

# Charting the Tractability Frontier of Certain Conjunctive Query Answering

Jef Wijsen  
Université de Mons, Belgium

## Abstract

An uncertain database is defined as a relational database in which primary keys need not be satisfied. A repair (or possible world) of such database is obtained by selecting a maximal number of tuples without ever selecting two distinct tuples with the same primary key value. For a Boolean query  $q$ , the decision problem  $\text{CERTAINTY}(q)$  takes as input an uncertain database  $\mathbf{db}$  and asks whether  $q$  is satisfied by every repair of  $\mathbf{db}$ . Our main focus is on acyclic Boolean conjunctive queries without self-join. Previous work [23] has introduced the notion of (directed) attack graph of such queries, and has proved that  $\text{CERTAINTY}(q)$  is first-order expressible if and only if the attack graph of  $q$  is acyclic. The current paper investigates the boundary between tractability and intractability of  $\text{CERTAINTY}(q)$ . We first classify cycles in attack graphs as either weak or strong, and then prove among others the following. If the attack graph of a query  $q$  contains a strong cycle, then  $\text{CERTAINTY}(q)$  is **coNP**-complete. If the attack graph of  $q$  contains no strong cycle and every weak cycle of it is terminal (i.e., no edge leads from a vertex in the cycle to a vertex outside the cycle), then  $\text{CERTAINTY}(q)$  is in **P**. We then partially address the only remaining open case, i.e., when the attack graph contains some nonterminal cycle and no strong cycle. Finally, we establish a relationship between the complexities of  $\text{CERTAINTY}(q)$  and evaluating  $q$  on probabilistic databases.

## 1 Introduction

Primary key violations are a natural way for modeling uncertainty in the relational model. If two distinct tuples have the same primary key value, then at least one of them must be mistaken, but we do not know which one. This representation of uncertainty is also used in probabilistic databases, where each tuple is associated with a probability and distinct tuples with the same primary key value are disjoint probabilistic events [18, page 35].

In this paper, the term *uncertain database* is used for databases with primary key constraints that need not be satisfied. A repair (or possible world) of an uncertain database  $\mathbf{db}$  is a maximal subset of  $\mathbf{db}$  that satisfies all primary key constraints. Semantics of querying follows the conventional paradigm of *consistent query answering* [2, 4]: Given a Boolean query  $q$ , the decision problem  $\text{CERTAINTY}(q)$  takes as input an uncertain database  $\mathbf{db}$  and asks whether  $q$  is satisfied by every repair of  $\mathbf{db}$ . Notice that  $q$  is not part of the input,

C	<u>conf</u>	<u>year</u>	<u>city</u>	R	<u>conf</u>	<u>rank</u>
	PODS	2016	Rome		PODS	A
	PODS	2016	Paris		KDD	A
	KDD	2017	Rome		KDD	B

Figure 1: Uncertain database.

so the complexity of the problem is data complexity. The restriction to Boolean queries simplifies the technical treatment, but is not fundamental.

Primary keys are underlined in the conference planning database of Fig. 1. Maximal sets of tuples that agree on their primary key, called *blocks*, are separated by dashed lines. There is uncertainty about the city of PODS 2016, and about the rank of KDD. The database has four repairs. The query  $\exists x \exists y (C(\underline{x}, y, \text{'Rome'}) \wedge R(\underline{x}, \text{'A'}))$  (Will Rome host some A conference?) is true in only three repairs.

The problem  $\text{CERTAINTY}(q)$  is in **coNP** for first-order queries  $q$  (a “no” certificate is a repair falsifying  $q$ ). Its complexity for conjunctive queries has attracted the attention of several authors, also outside the database community [5]. A major research objective is to find an effective method that takes as input a conjunctive query  $q$  and decides to which complexity classes  $\text{CERTAINTY}(q)$  belongs, or does not belong. Complexity classes of interest are the class of first-order expressible problems (or **AC**<sup>0</sup>), **P**, and **coNP**-complete.

Unless specified otherwise, whenever we say “query” in the remainder of this paper, we mean a Boolean conjunctive query without self-join (i.e., without repeated relation names). Such queries are called *acyclic* if they have a join tree [3].

Our previous work [21, 23] has revealed the frontier between first-order expressibility and inexpressibility of  $\text{CERTAINTY}(q)$  for acyclic queries  $q$ . In the current work, we study the frontier between tractability and intractability of  $\text{CERTAINTY}(q)$  for the same class of queries. That is, we aim at an effective method that takes as input a query  $q$  and decides whether  $\text{CERTAINTY}(q)$  is in **P** or **coNP**-complete (or neither of the two, which is theoretically possible if **P**  $\neq$  **coNP** [14]). For queries with exactly two atoms, such a method was recently found by Kolaitis and Pema [13], but moving from two to more than two atoms is a major challenge.

Uncertain databases become probabilistic by assuming that the probabilities of all repairs are equal and sum up to 1. In

probabilistic terms, distinct tuples of the same block represent disjoint (i.e., exclusive) events, while tuples of distinct blocks are independent. Such probabilistic databases have been called *block-independent-disjoint* (BID). The tractability/intractability frontier of query evaluation on BID probabilistic databases has been revealed by Dalvi et al. [8]. Here, evaluating a Boolean query is a function problem that takes as input a BID probabilistic database and asks the probability (a real number between 0 and 1) that  $q$  is true. The decision problem  $\text{CERTAINTY}(q)$ , on the other hand, simply asks whether this probability is equal to 1.

In previous work [23], we introduced the (directed) attack graph of an acyclic query, and showed that  $\text{CERTAINTY}(q)$  is first-order expressible if and only if  $q$ 's attack graph is acyclic. In the current paper, we study attack graphs in more depth. We will classify cycles in attack graphs as either weak or strong. The main contributions can then be summarized as follows.

1. If the attack graph of an acyclic query  $q$  contains a strong cycle, then  $\text{CERTAINTY}(q)$  is **coNP**-complete. This will be Theorem 2.
2. If the attack graph of an acyclic query  $q$  contains no strong cycle and all weak cycles of it are terminal (i.e., no edge leads from a vertex in the cycle to a vertex outside the cycle), then  $\text{CERTAINTY}(q)$  is in **P**. This will be Theorem 3.
3. The only acyclic queries  $q$  not covered by the two preceding results have an attack graph with some nonterminal cycle and without strong cycle. We provide supporting evidence for our conjecture that  $\text{CERTAINTY}(q)$  is tractable for such queries. Our results imply that  $\text{CERTAINTY}(q)$  is tractable for “cycle” queries  $q$  of the form  $\exists^*(R_1(\underline{x}_1, x_2) \wedge R_2(\underline{x}_2, x_3) \dots \wedge R_{k-1}(\underline{x}_{k-1}, x_k) \wedge R_k(\underline{x}_k, x_1))$ . These queries arise in the work of Fuxman and Miller [10]. The case  $k = 2$  was solved in [22], but the case  $k > 2$  was open and will be settled by Corollary 1.
4. Theorem 6 and its Corollary 2 will establish a relationship between the tractability frontiers of  $\text{CERTAINTY}(q)$  and query evaluation on probabilistic databases.

Our work significantly extends and generalizes known results in the literature.

The remainder of this paper is organized as follows. The next section further discusses related work. Section 3 defines the basic notions of certain conjunctive query answering. Section 4 defines the notion of attack graph. Sections 5 and 6 show our main intractability and tractability results respectively. Section 7 establishes a relationship between the complexities of  $\text{CERTAINTY}(q)$  and evaluating query  $q$  on probabilistic databases. Section 8 concludes the paper and raises challenges for future research. Several proofs have been moved to an Appendix.

## 2 More Related Work

The investigation of  $\text{CERTAINTY}(q)$  was pioneered by Fuxman and Miller [9, 10], who defined a class of queries  $q$  for which  $\text{CERTAINTY}(q)$  is first-order expressible. This class has later on been extended by Wijsen [21, 23], who developed an effective method to decide whether  $\text{CERTAINTY}(q)$  is first-order expressible for acyclic queries  $q$ . In their conclusion, Fuxman and Miller [9, 10] raised the question whether there exist queries  $q$ , without self-join, such that  $\text{CERTAINTY}(q)$  is in **P** but not first-order expressible. The first example of such a query was identified by Wijsen [22]. The current paper identifies a large class of such queries (all acyclic queries with a cyclic attack graph in which all cycles are weak and terminal).

Kolaitis and Pema [13] recently showed that for every query  $q$  with exactly two atoms,  $\text{CERTAINTY}(q)$  is either in **P** or **coNP**-complete, and it is decidable which of the two is the case. If  $\text{CERTAINTY}(q)$  is in **P** and not first-order expressible, then it can be reduced in polynomial time to the problem of finding maximal (with respect to cardinality) independent sets of vertices in claw-free graphs. The latter problem can be solved in polynomial time by an ingenious algorithm of Minty [17]. Unfortunately, the proposed reduction is not applicable on queries with more than two atoms.

The counting variant of  $\text{CERTAINTY}(q)$ , which has been denoted  $\sharp\text{CERTAINTY}(q)$ , takes as input an uncertain database **db** and asks to determine the number of repairs of **db** that satisfy query  $q$ . Maslowski and Wijsen [16, 15] have recently showed that for every query  $q$ , the counting problem  $\sharp\text{CERTAINTY}(q)$  is either in **FP** or  $\sharp\text{P}$ -complete, and it is decidable which of the two is the case.

As observed in Section 1, uncertain databases are a restricted case of block-independent-disjoint (BID) probabilistic databases [7, 8]. This observation will be elaborated in Section 7.

All aforementioned results assume queries without self-join. For queries  $q$  with self-joins, only fragmentary results about the complexity of  $\text{CERTAINTY}(q)$  are known [6, 20]. The extension to unions of conjunctive queries has been studied in [12].

## 3 Preliminaries

We assume disjoint sets of *variables* and *constants*. If  $\vec{x}$  is a sequence containing variables and constants, then  $\text{vars}(\vec{x})$  denotes the set of variables that occur in  $\vec{x}$ , and  $|\vec{x}|$  denotes the length of  $\vec{x}$ .

Let  $U$  be a set of variables. A *valuation over  $U$*  is a total mapping  $\theta$  from  $U$  to the set of constants. Such valuation  $\theta$  is extended to be the identity on constants and on variables not in  $U$ .

**Atoms and key-equal facts.** Every *relation name*  $R$  has a fixed *signature*, which is a pair  $[n, k]$  with  $n \geq k \geq 1$ : the integer  $n$  is the *arity* of the relation name and  $\{1, 2, \dots, k\}$  is the *primary key*. The relation name  $R$  is *all-key* if  $n = k$ . If  $R$  is a relation name with signature  $[n, k]$ , then  $R(s_1, \dots, s_n)$  is an *R-atom* (or simply atom), where each  $s_i$  is either a constant or a

variable ( $1 \leq i \leq n$ ). Such atom is commonly written as  $R(\underline{\vec{x}}, \vec{y})$  where the primary key value  $\vec{x} = s_1, \dots, s_k$  is underlined and  $\vec{y} = s_{k+1}, \dots, s_n$ . A *fact* is an atom in which no variable occurs. Two facts  $R_1(\underline{\vec{a}}_1, \vec{b}_1), R_2(\underline{\vec{a}}_2, \vec{b}_2)$  are *key-equal* if  $R_1 = R_2$  and  $\vec{a}_1 = \vec{a}_2$ .

We will use letters  $F, G, H, I$  for atoms, and  $A, B, C$  for facts of an uncertain database. For atom  $F = R(\underline{\vec{x}}, \vec{y})$ , we denote by  $\text{key}(F)$  the set of variables that occur in  $\vec{x}$ , and by  $\text{vars}(F)$  the set of variables that occur in  $F$ , that is,  $\text{key}(F) = \text{vars}(\vec{x})$  and  $\text{vars}(F) = \text{vars}(\vec{x}) \cup \text{vars}(\vec{y})$ .

**Uncertain database, blocks, and repairs.** A *database schema* is a finite set of *relation names*. All constructs that follow are defined relative to a fixed database schema.

An *uncertain database* is a finite set  $\mathbf{db}$  of facts using only the relation names of the schema. A *block* of  $\mathbf{db}$  is a maximal set of key-equal facts of  $\mathbf{db}$ . If  $A \in \mathbf{db}$ , then  $\text{block}(A, \mathbf{db})$  denotes the block of  $\mathbf{db}$  containing  $A$ . An uncertain database  $\mathbf{db}$  is *consistent* if it does not contain two distinct facts that are key-equal (i.e., if every block of  $\mathbf{db}$  is a singleton). A *repair* of  $\mathbf{db}$  is a maximal consistent subset of  $\mathbf{db}$ .<sup>1</sup>

**Boolean conjunctive query.** A *Boolean conjunctive query* is a finite set  $q = \{R_1(\underline{\vec{x}}_1, \vec{y}_1), \dots, R_n(\underline{\vec{x}}_n, \vec{y}_n)\}$  of atoms. By  $\text{vars}(q)$ , we denote the set of variables that occur in  $q$ . The set  $q$  represents the first-order sentence

$$\exists u_1 \dots \exists u_k (R_1(\underline{\vec{x}}_1, \vec{y}_1) \wedge \dots \wedge R_n(\underline{\vec{x}}_n, \vec{y}_n)),$$

where  $\{u_1, \dots, u_k\} = \text{vars}(q)$ . The query  $q$  is *satisfied* by uncertain database  $\mathbf{db}$ , denoted  $\mathbf{db} \models q$ , if there exists a valuation  $\theta$  over  $\text{vars}(q)$  such that for each  $i \in \{1, \dots, n\}$ ,  $R_i(\theta(\underline{\vec{x}}_i), \theta(\vec{y}_i)) \in \mathbf{db}$ . We say that  $q$  has a *self-join* if some relation name occurs more than once in  $q$  (i.e., if  $R_i = R_j$  for some  $1 \leq i < j \leq n$ ).

The restriction to Boolean queries simplifies the technical treatment, but is not fundamental. Since every relation name has a fixed signature, relevant primary key constraints are implicitly present in all queries; moreover, primary keys will be underlined.

**Join tree and acyclic conjunctive query.** The notions of join tree and acyclicity [3] are recalled next. A *join tree* for a conjunctive query  $q$  is an undirected tree whose vertices are the atoms of  $q$  such that the following condition is satisfied:

*Connectedness Condition.* Whenever the same variable  $x$  occurs in two atoms  $F$  and  $G$ , then  $x$  occurs in each atom on the unique path linking  $F$  and  $G$ .

Commonly, an edge between atoms  $F$  and  $G$  is labeled by the (possibly empty) set  $\text{vars}(F) \cap \text{vars}(G)$ . The term *Connectedness Condition* appears in [11] and refers to the fact that the set of vertices in which  $x$  occurs induces a connected subtree. A conjunctive query  $q$  is *acyclic* if it has a join tree. The symbol  $\tau$  will be used for join trees. We write  $F \overset{L}{\sim} G$  to denote an

edge between  $F$  and  $G$  with label  $L$ . A join tree is shown in Fig. 2 (left).

**Certain query answering.** Given a Boolean conjunctive query  $q$ ,  $\text{CERTAINTY}(q)$  is (the complexity of) the following set.

$$\text{CERTAINTY}(q) = \{\mathbf{db} \mid \mathbf{db} \text{ is an uncertain database such that every repair of } \mathbf{db} \text{ satisfies } q\}$$

$\text{CERTAINTY}(q)$  is said to be *first-order expressible* if there exists a first-order sentence  $\phi$  such that for every uncertain database  $\mathbf{db}$ ,  $\mathbf{db} \in \text{CERTAINTY}(q)$  if and only if  $\mathbf{db} \models \phi$ . The formula  $\phi$ , if it exists, is called a *certain first-order rewriting* of  $q$ .

**Purified uncertain databases.** Let  $q$  be a Boolean conjunctive query. An uncertain database  $\mathbf{db}$  is said to be *purified relative to  $q$*  if for every fact  $A \in \mathbf{db}$ , there exists a valuation  $\theta$  over  $\text{vars}(q)$  such that  $A \in \theta(q) \subseteq \mathbf{db}$ . Intuitively, every fact in a purified uncertain database is relevant for the query. This notion of purified database is new and illustrated next.

**Example 1** The uncertain database  $\{R(\underline{a}, b), S(\underline{b}, a), S(\underline{b}, c)\}$  is not purified relative to query  $\{R(\underline{x}, y), S(\underline{y}, x)\}$  because it contains no  $R$ -fact that “joins” with  $S(\underline{b}, c)$ .  $\triangleleft$

The following lemma implies that in the study of tractability of  $\text{CERTAINTY}(q)$ , we can assume without loss of generality that uncertain databases are purified; this assumption will simplify the technical treatment. Notice that the query  $q$  in the lemma’s statement is not required to be acyclic.

**Lemma 1** Let  $q$  be a Boolean conjunctive query. Let  $\mathbf{db}_0$  be an uncertain database. It is possible to compute in polynomial time an uncertain database  $\mathbf{db}$  that is purified relative to  $q$  such that

$$\mathbf{db} \in \text{CERTAINTY}(q) \iff \mathbf{db}_0 \in \text{CERTAINTY}(q).$$

## 4 Attack Graph

The primary key of an atom  $F$  gives rise to a functional dependency among the variables that occur in  $F$ . For example,  $R(\underline{x}, y, z, u)$  gives rise to  $\{x, y\} \rightarrow \{x, y, z, u\}$ , which will be abbreviated as  $xy \rightarrow xyzu$  (and which is equivalent to  $xy \rightarrow zu$ ). The set  $\mathcal{K}(q)$  defined next collects all functional dependencies that arise in atoms of  $q$ .

**Definition 1** Let  $q$  be a Boolean conjunctive query. We define  $\mathcal{K}(q)$  as the following set of functional dependencies.

$$\mathcal{K}(q) = \{\text{key}(F) \rightarrow \text{vars}(F) \mid F \in q\} \quad \triangleleft$$

Concerning the following definition, recall from relational database theory [19, page 387] that if  $\Sigma$  is a set of functional dependencies over a set  $U$  of attributes and  $X \subseteq U$ , then the attribute closure of  $X$  (with respect to  $\Sigma$ ) is the set  $\{A \in U \mid \Sigma \models X \rightarrow A\}$ .

**Definition 2** Let  $q$  be a Boolean conjunctive query. For every  $F \in q$ , we define  $F^{+,q}$  as the following set of variables.

$$F^{+,q} = \{x \in \text{vars}(q) \mid \mathcal{K}(q \setminus \{F\}) \models \text{key}(F) \rightarrow x\} \quad \triangleleft$$

<sup>1</sup>It makes no difference whether the word “maximal” refers to cardinality of sets or set-containment.

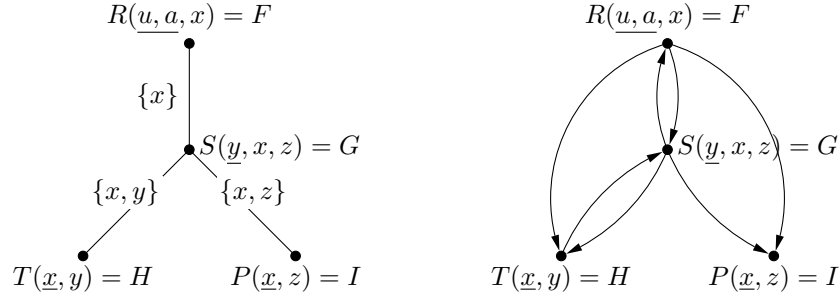


Figure 2: Join tree (left) and attack graph (right) of query  $q_1$ . The attack from  $G$  to  $F$  is strong. All other attacks are weak.

In words,  $F^{+,q}$  is the attribute closure of the set  $\text{key}(F)$  with respect to the set of functional dependencies that arise in the atoms of  $q \setminus \{F\}$ . Note that variables play the role of attributes in our framework.

**Example 2** Let  $q_1 = \{R(u, a, x), S(y, x, z), T(x, y), P(x, z)\}$ . A join tree for this query is shown in Fig. 2 (left). To shorten notation, let  $F = R(u, a, x)$ ,  $G = S(y, x, z)$ ,  $H = T(x, y)$ , and  $I = P(x, z)$ , as indicated in the figure. We have the following.

$$\mathcal{K}(q_1 \setminus \{F\}) = \{y \rightarrow xyz, x \rightarrow xy, x \rightarrow xz\}$$

$$\text{key}(F) = \{u\} \text{ and } F^{+,q_1} = \{u\}$$

$$\mathcal{K}(q_1 \setminus \{G\}) = \{u \rightarrow ux, x \rightarrow xy, x \rightarrow xz\}$$

$$\text{key}(G) = \{y\} \text{ and } G^{+,q_1} = \{y\}$$

$$\mathcal{K}(q_1 \setminus \{H\}) = \{u \rightarrow ux, y \rightarrow xyz, x \rightarrow xz\}$$

$$\text{key}(H) = \{x\} \text{ and } H^{+,q_1} = \{x, z\}$$

$$\mathcal{K}(q_1 \setminus \{I\}) = \{u \rightarrow ux, y \rightarrow xyz, x \rightarrow xy\}$$

$$\text{key}(I) = \{x\} \text{ and } I^{+,q_1} = \{x, y, z\}$$

◁

**Definition 3** Let  $q$  be an acyclic Boolean conjunctive query. Let  $\tau$  be a join tree for  $q$ . The *attack graph* of  $\tau$  is a directed graph whose vertices are the atoms of  $q$ . There is a directed edge from  $F$  to  $G$  if  $F, G$  are distinct atoms such that for every label  $L$  on the unique path that links  $F$  and  $G$  in  $\tau$ , we have  $L \not\subseteq F^{+,q}$ .

We write  $F \xrightarrow{\tau} G$  if the attack graph of  $\tau$  contains a directed edge from  $F$  to  $G$ . The directed edge  $F \xrightarrow{\tau} G$  is also called an *attack from  $F$  to  $G$* . If  $F \xrightarrow{\tau} G$ , we say that  $F$  *attacks*  $G$  (or that  $G$  is attacked by  $F$ ). ◁

**Example 3** This is a continuation of Example 2. Fig. 2 (left) shows a join tree  $\tau_1$  for query  $q_1$ . The attack graph of  $\tau_1$  is shown in Fig. 2 (right) and is computed as follows.

Let us first compute the attacks outgoing from  $F$ . The path from  $F$  to  $G$  in the join tree is  $F \xrightarrow{\{x\}} G$ . Since the label  $\{x\}$  is not contained in  $F^{+,q_1}$ , the attack graph contains a directed edge from  $F$  to  $G$ , i.e.,  $F \xrightarrow{\tau_1} G$ . The path from  $F$  to  $H$  in the join tree is  $F \xrightarrow{\{x\}} G \xrightarrow{\{x,y\}} H$ . Since no label on that path is contained in  $F^{+,q_1}$ , the attack graph contains a directed edge from  $F$  to  $H$ . In the same way, one finds that  $F$  attacks  $I$ .

Let us next compute the attacks outgoing from  $H$ . The path from  $H$  to  $G$  in the join tree is  $H \xrightarrow{\{x,y\}} G$ . Since the label  $\{x, y\}$  is not contained in  $G^{+,q_1}$ , the attack graph contains a directed edge from  $H$  to  $G$ , i.e.,  $H \xrightarrow{\tau_1} G$ . The path from  $H$  to  $F$  in the join tree is  $H \xrightarrow{\{x,y\}} G \xrightarrow{\{x\}} F$ . Since the label  $\{x\}$  is contained in  $H^{+,q_1}$ , the attack graph contains no directed edge from  $H$  to  $F$ . And so on. The complete attack graph is shown in Fig. 2 (right). ◁

Remarkably, it was shown in [23] that if  $\tau_1$  and  $\tau_2$  are distinct join trees for the same acyclic query  $q$ , then the attack graph of  $\tau_1$  is identical to the attack graph of  $\tau_2$ . This motivates the following definition.

**Definition 4** Let  $q$  be an acyclic Boolean conjunctive query. The attack graph of  $q$  is the attack graph of  $\tau$  for any join tree  $\tau$  for  $q$ . We write  $F \xrightarrow{q} G$  (or simply  $F \rightsquigarrow G$  if  $q$  is clear from the context) to indicate that the attack graph of  $q$  contains a directed edge from  $F$  to  $G$ . We write  $F \not\xrightarrow{q} G$  if it is not the case that  $F \xrightarrow{q} G$ . ◁

The attack graph of an acyclic query  $q$  can be computed in quadratic time in the length of  $q$  [23]. Figures 4 and 5 show attack graphs, but omit join trees. The main result in [23] is the following.

**Theorem 1 ([23])** *The following are equivalent for all acyclic Boolean conjunctive queries  $q$  without self-join:*

1. The attack graph of  $q$  is acyclic.
2. CERTAINTY( $q$ ) is first-order expressible.

Finally, we provide two lemmas that will be useful later on.

**Lemma 2** *Let  $q$  be an acyclic Boolean conjunctive query. Let  $F, G$  be distinct atoms of  $q$ . If  $F \rightsquigarrow G$ , then  $\text{key}(G) \not\subseteq F^{+,q}$  and  $\text{vars}(F) \not\subseteq F^{+,q}$ .*

**Lemma 3 ([23])** *Let  $q$  be an acyclic Boolean conjunctive query. Let  $F, G, H$  be distinct atoms of  $q$ . If  $F \rightsquigarrow G$  and  $G \rightsquigarrow H$ , then  $F \rightsquigarrow H$  or  $G \rightsquigarrow F$ .*

## 5 Intractability

The following definition classifies cycles in attack graphs as either strong or weak. The main result of this section is that  $\text{CERTAINTY}(q)$  is **coNP**-complete for acyclic queries  $q$  whose attack graph contains a strong cycle.

**Definition 5** Let  $q$  be an acyclic Boolean conjunctive query. For every  $F \in q$ , we define  $F^{\boxplus,q}$  as the following set of variables.

$$F^{\boxplus,q} = \{x \in \text{vars}(q) \mid \mathcal{K}(q) \models \text{key}(F) \rightarrow x\}$$

An attack  $F \rightsquigarrow G$  in the attack graph of  $q$  is called *weak* if  $\text{key}(G) \subseteq F^{\boxplus,q}$ . An attack that is not weak, is called *strong*.

A (directed) *cycle of size  $n$*  in the attack graph of  $q$  is a sequence of edges  $F_0 \rightsquigarrow F_1 \rightsquigarrow F_2 \dots \rightsquigarrow F_{n-1} \rightsquigarrow F_0$  such that  $i \neq j$  implies  $F_i \neq F_j$ . Thus, *cycle* means elementary cycle.

A cycle in the attack graph of  $q$  is called *strong* if at least one attack in the cycle is strong. A cycle that is not strong, is called *weak*.  $\triangleleft$

It is straightforward that  $F^{+,q} \subseteq F^{\boxplus,q}$ .

**Example 4** For the query  $q_1$  in Fig. 2, we have the following.

$$\begin{aligned} \mathcal{K}(q_1) &= \{u \rightarrow ux, y \rightarrow xyz, x \rightarrow xy, x \rightarrow xz\} \\ F^{\boxplus,q_1} &= \{u, x, y, z\} \\ G^{\boxplus,q_1} &= \{x, y, z\} \\ H^{\boxplus,q_1} &= \{x, y, z\} \\ I^{\boxplus,q_1} &= \{x, y, z\} \end{aligned}$$

The attack  $F \xrightarrow{q_1} G$  is weak, because  $\text{key}(G) = \{x\} \subseteq F^{\boxplus,q_1}$ . The attack  $G \xrightarrow{q_1} F$  is strong, because  $\text{key}(F) = \{u\} \not\subseteq G^{\boxplus,q_1}$ . One can verify that the attack from  $G$  to  $F$  is the only strong attack in the attack graph of  $q_1$ .

The attack cycle  $G \xrightarrow{q_1} H \xrightarrow{q_1} G$  is weak. The attack cycle  $F \xrightarrow{q_1} G \xrightarrow{q_1} F$  is strong, because it contains the strong attack  $G \xrightarrow{q_1} F$ . For the same reason, the attack cycle  $F \xrightarrow{q_1} H \xrightarrow{q_1} G \xrightarrow{q_1} F$  is strong.  $\triangleleft$

Example 4 showed that the attack graph of  $q_1$  has a strong cycle of length 3, and a strong cycle of length 2. This is no coincidence, as stated by the following lemma.

**Lemma 4** Let  $q$  be an acyclic Boolean conjunctive query. If the attack graph of  $q$  contains a strong cycle, then it contains a strong cycle of length 2.

The following proof establishes that for every acyclic query  $q$  whose attack graph contains a strong cycle, there exists a polynomial-time many-one reduction from  $\text{CERTAINTY}(q_0)$  to  $\text{CERTAINTY}(q)$ , where  $q_0 = \{R_0(\underline{x}, y), S_0(y, z, x)\}$ . Since  $\text{CERTAINTY}(q_0)$  was proved **coNP**-hard by Kolaitis and Pema [13], we obtain the desired **coNP**-hard lower bound for  $\text{CERTAINTY}(q)$ . As the proof is rather involved, we provide in Fig. 3 a mnemonic for the construction in the beginning of the proof. To further improve readability, some parts of the proof will be stated as sublemmas.

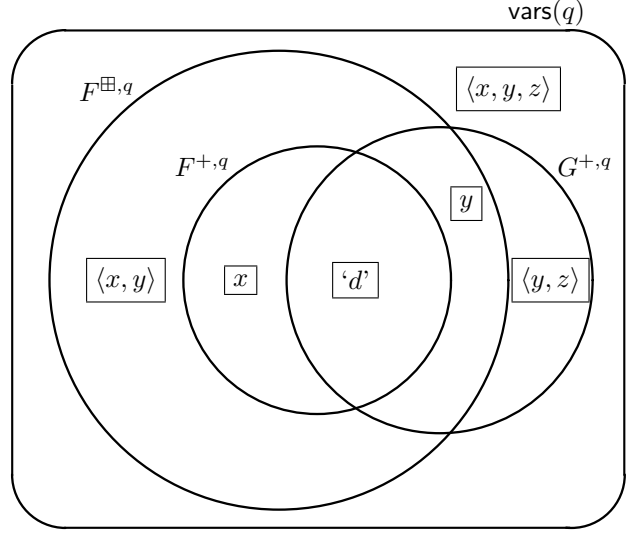


Figure 3: Help for the proof of Theorem 2.

**Theorem 2** Let  $q$  be an acyclic Boolean conjunctive query without self-join. If the attack graph of  $q$  contains a strong cycle, then  $\text{CERTAINTY}(q)$  is **coNP**-complete.

**Proof** Since  $\text{CERTAINTY}(q)$  is obviously in **coNP**, it suffices to show that it is **coNP**-hard. Assume that the attack graph of  $q$  contains a strong cycle. By Lemma 4, we can assume  $F, G \in q$  such that  $F \rightsquigarrow G \rightsquigarrow F$  and the attack  $F \rightsquigarrow G$  is strong. For every valuation  $\theta$  over  $\{x, y, z\}$ , we define  $\hat{\theta}$  as the following valuation over  $\text{vars}(q)$ .

1. If  $u \in F^{+,q} \cap G^{+,q}$ , then  $\hat{\theta}(u) = 'd'$  for some fixed constant  $d$ ;
2. if  $u \in F^{+,q} \setminus G^{+,q}$ , then  $\hat{\theta}(u) = \theta(x)$ ;
3. if  $u \in G^{+,q} \setminus F^{\boxplus,q}$ , then  $\hat{\theta}(u) = \langle \theta(y), \theta(z) \rangle$ ;
4. if  $u \in (G^{+,q} \cap F^{\boxplus,q}) \setminus F^{+,q}$ , then  $\hat{\theta}(u) = \theta(y)$ ;
5. if  $u \in F^{\boxplus,q} \setminus (F^{+,q} \cup G^{+,q})$ , then  $\hat{\theta}(u) = \langle \theta(x), \theta(y) \rangle$ ; and
6. if  $u \notin F^{\boxplus,q} \cup G^{+,q}$ , then  $\hat{\theta}(u) = \langle \theta(x), \theta(y), \theta(z) \rangle$ .

Notice that  $\hat{\theta}(u)$  can be a sequence of length two or three; two sequences of the same length are equal if they contain the same elements in the same order. The Venn diagram of Fig. 3 will come in handy: every region contains a boxed label that indicates how  $\hat{\theta}(u)$  is computed for variables  $u$  in that region. For example, assume  $u$  belongs to the region with label  $\langle x, y \rangle$  (i.e.,  $u \in F^{\boxplus,q} \setminus (F^{+,q} \cup G^{+,q})$ ), then  $\hat{\theta}(u) = \langle \theta(x), \theta(y) \rangle$ .

We show three sublemmas that will be used later on in the proof.

**Sublemma 1** Let  $\theta_1, \theta_2$  be two valuations over  $\{x, y, z\}$ . If  $H \in q$  such that  $F \neq H \neq G$ , then  $\{\hat{\theta}_1(H), \hat{\theta}_2(H)\}$  is consistent.

**Proof Sublemma 1** Let  $H \in q$  such that  $F \neq H \neq G$ . Assume the following.

$$\text{For every } u \in \text{key}(H), \hat{\theta}_1(u) = \hat{\theta}_2(u). \quad (1)$$

It suffices to show the following.

$$\text{For every } u \in \text{vars}(H), \hat{\theta}_1(u) = \hat{\theta}_2(u). \quad (2)$$

We consider four cases.

**Case  $\theta_1(x) = \theta_2(x)$  and  $\theta_1(y) = \theta_2(y)$ .** If  $\theta_1(z) = \theta_2(z)$ , then  $\theta_1 = \theta_2$ , and (2) holds vacuously. Assume next  $\theta_1(z) \neq \theta_2(z)$ . Then it follows from (1) that no variable of  $\text{key}(H)$  belongs to a region of the Venn diagram (see Fig. 3) that contains  $z$ . Since  $z$  occurs in all regions outside  $F^{\boxplus,q}$ , we conclude  $\text{key}(H) \subseteq F^{\boxplus,q}$ . Since  $\mathcal{K}(q)$  contains  $\text{key}(H) \rightarrow \text{vars}(H)$ , it follows  $\text{vars}(H) \subseteq F^{\boxplus,q}$ . Since  $z$  does not occur inside  $F^{\boxplus,q}$  in the Venn diagram, we conclude (2).

**Case  $\theta_1(x) = \theta_2(x)$  and  $\theta_1(y) \neq \theta_2(y)$ .** By (1), no variable of  $\text{key}(H)$  belongs to a region of the Venn diagram that contains  $y$ . It follows  $\text{key}(H) \subseteq F^{+,q}$ . Consequently,  $\text{vars}(H) \subseteq F^{+,q}$ . Since neither  $y$  nor  $z$  occurs inside  $F^{+,q}$  in the Venn diagram, we conclude (2).

**Case  $\theta_1(x) \neq \theta_2(x)$  and  $\theta_1(y) = \theta_2(y)$ .** First assume  $\theta_1(z) = \theta_2(z)$ . By (1), no variable of  $\text{key}(H)$  belongs to a region of the Venn diagram that contains  $x$ . Consequently,  $\text{key}(H) \subseteq G^{+,q}$ . It follows  $\text{vars}(H) \subseteq G^{+,q}$ . Since  $x$  does not occur inside  $G^{+,q}$  in the Venn diagram, we conclude (2).

Next assume  $\theta_1(z) \neq \theta_2(z)$ . By (1), no variable of  $\text{key}(H)$  belongs to a region of the Venn diagram that contains  $x$  or  $z$ . Consequently,  $\text{key}(H) \subseteq F^{\boxplus,q} \cap G^{+,q}$ . It follows  $\text{vars}(H) \subseteq F^{\boxplus,q} \cap G^{+,q}$ . Since neither  $x$  nor  $z$  occurs inside  $F^{\boxplus,q} \cap G^{+,q}$  in the Venn diagram, we conclude (2).

**Case  $\theta_1(x) \neq \theta_2(x)$  and  $\theta_1(y) \neq \theta_2(y)$ .** By (1), no variable of  $\text{key}(H)$  belongs to a region of the Venn diagram that contains  $x$  or  $y$ . Consequently,  $\text{key}(H) \subseteq F^{+,q} \cap G^{+,q}$ . It follows  $\text{vars}(H) \subseteq F^{+,q} \cap G^{+,q}$ . Since none of  $x, y$ , or  $z$  occurs inside  $F^{+,q} \cap G^{+,q}$  in the Venn diagram, we conclude (2). This concludes the proof of Sublemma 1.  $\square$

**Sublemma 2** Let  $\theta_1, \theta_2$  be two valuations over  $\{x, y, z\}$ .

1.  $\hat{\theta}_1(F)$  and  $\hat{\theta}_2(F)$  are key-equal  $\iff \theta_1(x) = \theta_2(x)$ .
2.  $\hat{\theta}_1(F) = \hat{\theta}_2(F) \iff \theta_1(x) = \theta_2(x) \text{ and } \theta_1(y) = \theta_2(y)$ .

**Sublemma 3** Let  $\theta_1, \theta_2$  be two valuations over  $\{x, y, z\}$ .

1.  $\hat{\theta}_1(G)$  and  $\hat{\theta}_2(G)$  are key-equal  $\iff \theta_1(y) = \theta_2(y)$  and  $\theta_1(z) = \theta_2(z)$ .
2.  $\hat{\theta}_1(G) = \hat{\theta}_2(G) \iff \theta_1 = \theta_2$ .

We continue the proof of Theorem 2. Let  $q_0 = \{R_0(\underline{x}, y), S_0(y, z, x)\}$ . The signatures of  $R_0$  and  $S_0$  are  $[2, 1]$  and  $[3, 2]$

respectively. Let  $F_0 = R_0(\underline{x}, y)$  and  $G_0 = S_0(y, z, x)$ . In the remainder of the proof, we establish a polynomial-time many-one reduction from  $\text{CERTAINTY}(q_0)$  to  $\text{CERTAINTY}(q)$ . **coNP**-hardness of  $\text{CERTAINTY}(q)$  then follows from **coNP**-hardness of  $\text{CERTAINTY}(q_0)$ , which was established in [13].

Let  $\mathbf{db}_0$  be an uncertain database. By Lemma 1, we can assume that  $\mathbf{db}_0$  is purified relative to  $q_0$ . Let  $\mathcal{V}$  be the set of valuations  $\theta$  over  $\{x, y, z\}$  such that  $\theta(q_0) \subseteq \mathbf{db}_0$ . Since  $\mathbf{db}_0$  is purified, the following holds.

$$\mathbf{db}_0 = \{\theta(F_0) \mid \theta \in \mathcal{V}\} \cup \{\theta(G_0) \mid \theta \in \mathcal{V}\}$$

Let  $\mathbf{db} = \{\hat{\theta}(H) \mid H \in q, \theta \in \mathcal{V}\}$ . Since  $\mathcal{V}$  can be computed in polynomial time in the size of  $\mathbf{db}_0$ , the reduction from  $\mathbf{db}_0$  to  $\mathbf{db}$  is in polynomial time. Since  $q$  contains no self-join, the set  $\mathbf{db}$  is partitioned by the three disjoint subsets defined next.

$$\mathbf{db}_F = \{\hat{\theta}(F) \mid \theta \in \mathcal{V}\}$$

$$\mathbf{db}_G = \{\hat{\theta}(G) \mid \theta \in \mathcal{V}\}$$

$$\mathbf{db}_{\text{rest}} = \{\hat{\theta}(H) \mid H \in q, F \neq H \neq G, \theta \in \mathcal{V}\}$$

Since  $\mathbf{db}_{\text{rest}}$  is consistent by Sublemma 1, every repair of  $\mathbf{db}$  is the disjoint union of  $\mathbf{db}_{\text{rest}}$ , a repair of  $\mathbf{db}_F$ , and a repair of  $\mathbf{db}_G$ . In the next step of the proof, we establish a one-to-one relationship between repairs of  $\mathbf{db}_0$  and repairs of  $\mathbf{db}$ .

The function map will map repairs of  $\mathbf{db}_0$  to repairs of  $\mathbf{db}$ . For every repair  $\mathbf{r}_0$  of  $\mathbf{db}_0$ ,  $\text{map}(\mathbf{r}_0)$  is the disjoint union of three sets, as follows.

$$\begin{aligned} \text{map}(\mathbf{r}_0) &= \{\hat{\theta}(F) \mid \theta(F_0) \in \mathbf{r}_0, \theta \in \mathcal{V}\} \\ &\cup \{\hat{\theta}(G) \mid \theta(G_0) \in \mathbf{r}_0, \theta \in \mathcal{V}\} \\ &\cup \mathbf{db}_{\text{rest}} \end{aligned}$$

Clearly, the first of these three sets is contained in  $\mathbf{db}_F$ , and the second in  $\mathbf{db}_G$ . By Sublemmas 2 and 3, for every  $\theta \in \mathcal{V}$ ,

$$\theta(F_0) \in \mathbf{r}_0 \iff \hat{\theta}(F) \in \text{map}(\mathbf{r}_0) \quad (3)$$

$$\theta(G_0) \in \mathbf{r}_0 \iff \hat{\theta}(G) \in \text{map}(\mathbf{r}_0) \quad (4)$$

To prove the  $\Leftarrow$ -direction of (3) (the other implications are straightforward), assume  $A \in \text{map}(\mathbf{r}_0)$  with  $A = \hat{\theta}(F)$ . By the definition of map, we can assume  $\theta' \in \mathcal{V}$  such that  $\theta'(F_0) \in \mathbf{r}_0$  and  $\hat{\theta}'(F) = A$ . From  $\hat{\theta}(F) = \hat{\theta}'(F)$ , it follows by Sublemma 2 that  $\theta(F_0) = \theta'(F_0)$ , hence  $\theta(F_0) \in \mathbf{r}_0$ .

The following sublemma states that map is a bijection from the set of repairs of  $\mathbf{db}_0$  to the set of repairs of  $\mathbf{db}$ .

**Sublemma 4** 1. If  $\mathbf{r}_0$  is a repair of  $\mathbf{db}_0$ , then  $\text{map}(\mathbf{r}_0)$  is a repair of  $\mathbf{db}$ .

2. For every repair  $\mathbf{r}$  of  $\mathbf{db}$ , there exists a repair  $\mathbf{r}_0$  of  $\mathbf{db}_0$  such that  $\mathbf{r} = \text{map}(\mathbf{r}_0)$ .
3. If  $\mathbf{r}_0, \mathbf{r}'_0$  are distinct repairs of  $\mathbf{db}_0$ , then  $\text{map}(\mathbf{r}_0) \neq \text{map}(\mathbf{r}'_0)$ .

To conclude the proof of Theorem 2, we show:

$$\mathbf{db}_0 \in \text{CERTAINTY}(q_0) \iff \mathbf{db} \in \text{CERTAINTY}(q).$$

By Sublemma 4, it is sufficient to prove that for every repair  $\mathbf{r}_0$  of  $\mathbf{db}_0$ ,

$$\mathbf{r}_0 \models q_0 \iff \text{map}(\mathbf{r}_0) \models q.$$

$\implies$  Assume  $\mathbf{r}_0 \models q_0$ . We can assume  $\theta \in \mathcal{V}$  such that  $\theta(q_0) \subseteq \mathbf{r}_0$ . Obviously,  $\widehat{\theta}(q) \subseteq \text{map}(\mathbf{r}_0)$ .

$\impliedby$  Assume  $\text{map}(\mathbf{r}_0) \models q$ . We can assume a valuation  $\mu$  over  $\text{vars}(q)$  such that  $\mu(q) \subseteq \text{map}(\mathbf{r}_0)$ .

Let  $\tau$  be a join tree for  $q$ . Let  $H_0 \overset{L_1}{\sim} H_1 \dots \overset{L_\ell}{\sim} H_\ell$  be the unique path in  $\tau$  between  $F$  and  $G$ , where  $H_0 = F$  and  $H_\ell = G$ . For  $i \in \{0, \dots, \ell\}$ , we can assume  $\theta_i \in \mathcal{V}$  such that  $\mu(H_i) = \widehat{\theta}_i(H_i) \in \text{map}(\mathbf{r}_0)$ . Let  $i \in \{0, \dots, \ell-1\}$ . We show  $\theta_i(x) = \theta_{i+1}(x)$  and  $\theta_i(y) = \theta_{i+1}(y)$ . Since  $F \rightsquigarrow G \rightsquigarrow F$ , the label  $L_i$  contains a variable  $u_i$  such that  $u_i \notin F^{+,q}$  and a variable  $w_i$  such that  $w_i \notin G^{+,q}$  (possibly  $u_i = w_i$ ).

Since  $u_i \in \text{vars}(H_i) \cap \text{vars}(H_{i+1})$ , it must be the case that  $\widehat{\theta}_i(u_i) = \mu(u_i) = \widehat{\theta}_{i+1}(u_i)$ . Since  $y$  occurs in every region outside  $F^{+,q}$  in the Venn diagram (Fig. 3) and  $u_i \notin F^{+,q}$ , it is correct to conclude  $\theta_i(y) = \theta_{i+1}(y)$ .

Likewise, since  $w_i \in \text{vars}(H_i) \cap \text{vars}(H_{i+1})$ , it must be the case that  $\widehat{\theta}_i(w_i) = \mu(w_i) = \widehat{\theta}_{i+1}(w_i)$ . Since  $x$  occurs in every region outside  $G^{+,q}$  in the Venn diagram and  $w_i \notin G^{+,q}$ , it is correct to conclude  $\theta_i(x) = \theta_{i+1}(x)$ .

Consequently,  $\theta_0(x) = \theta_\ell(x)$  and  $\theta_0(y) = \theta_\ell(y)$ . From  $\widehat{\theta}_0(H_0), \widehat{\theta}_\ell(H_\ell) \in \text{map}(\mathbf{r}_0)$ ,  $H_0 = F$ , and  $H_\ell = G$ , it follows  $\theta_0(F_0), \theta_\ell(G_0) \in \mathbf{r}_0$  by (3) and (4). Since  $\theta_0$  and  $\theta_\ell$  agree on each variable in  $\text{vars}(F_0) \cap \text{vars}(G_0) = \{x, y\}$ , it follows  $\mathbf{r}_0 \models q_0$ . This concludes the proof of Theorem 2.  $\square$

## 6 Tractability

We conjecture that if the attack graph of an acyclic query  $q$  contains no strong cycle, then  $\text{CERTAINTY}(q)$  is in **P**.

**Conjecture 1** *Let  $q$  be an acyclic Boolean conjunctive query without self-join. If all cycles in the attack graph of  $q$  are weak, then  $\text{CERTAINTY}(q)$  is in **P**.*

Notice that by Theorem 1, we know that Conjecture 1 holds in the special case where  $q$ 's attack graph contains no cycle at all. Theorem 2 and Conjecture 1 together imply that for every acyclic query  $q$ ,  $\text{CERTAINTY}(q)$  is either in **P** or **coNP**-complete. In the following section, a somewhat weaker version of Conjecture 1 is proved.

### 6.1 All Cycles are Weak and Terminal

We show a weaker version of Conjecture 1. In this weaker version, the premise ‘‘all cycles are weak’’ is strengthened into ‘‘all cycles are weak and terminal.’’

**Definition 6** A cycle in a directed graph is called *terminal* if the graph contains no directed edge from a vertex in the cycle to a vertex outside the cycle. A cycle is *nonterminal* if it is not terminal.  $\triangleleft$

**Example 5** Figure 4 shows the attack graph of the acyclic query  $\{R_1(x, u_1, u_2, z), R_2(x, u_2, u_1, z), R_3(x, y, u_3, u_4),$

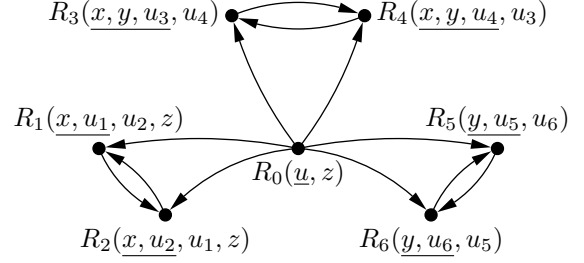


Figure 4: Attack graph. All cycles are weak and terminal.

$R_4(x, y, u_4, u_3), R_5(y, u_5, u_6), R_6(y, u_6, u_5)\}$ . All attack cycles are terminal and weak.  $\triangleleft$

**Example 6** In the attack graph of Fig. 5, all cycles are weak, but no cycle is terminal.  $\triangleleft$

**Theorem 3** *Let  $q$  be an acyclic Boolean conjunctive query without self-join. If all cycles in the attack graph of  $q$  are weak and terminal, then  $\text{CERTAINTY}(q)$  is in **P**.*

Notice that if a query  $q$  has exactly two atoms, then  $q$  is acyclic and every cycle in  $q$ 's attack graph must be terminal. Therefore Theorems 2 and 3 together imply the dichotomy theorem of Kolaitis and Pema [13].

To prove Theorem 3, we need four helping lemmas. In simple words, the first lemma states that if we replace a variable with a constant in an acyclic query, then no new attacks are generated, and weak attacks cannot become strong.

**Definition 7** Let  $q$  be a Boolean conjunctive query. If  $\vec{x} = \langle x_1, \dots, x_\ell \rangle$  is a sequence of distinct variables and  $\vec{a} = \langle a_1, \dots, a_\ell \rangle$  a sequence of constants, then  $q[\vec{x} \rightarrow \vec{a}]$  denotes the query obtained from  $q$  by replacing each occurrence of  $x_i$  with  $a_i$ , for all  $i \in \{1, \dots, \ell\}$ . If  $\theta$  is a valuation, then  $\theta[\vec{x} \rightarrow \vec{a}]$  is the valuation such that  $\theta[\vec{x} \rightarrow \vec{a}](\vec{x}) = \vec{a}$  and  $\theta[\vec{x} \rightarrow \vec{a}](y) = \theta(y)$  if  $y \notin \text{vars}(\vec{x})$ .  $\triangleleft$

**Lemma 5** *Let  $q$  be an acyclic Boolean conjunctive query without self-join. Let  $F, G \in q$ . Let  $z \in \text{vars}(q)$  and let  $c$  be a constant. Let  $q' = q[z \rightarrow c]$ ,  $F' = F[z \rightarrow c]$ , and  $G' = G[z \rightarrow c]$ . Then, the following hold.*

1.  $q'$  is acyclic.
2. If  $F' \overset{q'}{\rightsquigarrow} G'$ , then  $F \overset{q}{\rightsquigarrow} G$ .
3. If  $F' \overset{q'}{\rightsquigarrow} G'$  and  $F \overset{q}{\rightsquigarrow} G$  is a weak attack, then  $F' \overset{q'}{\rightsquigarrow} G'$  is a weak attack.

**Lemma 6** *Let  $q$  be an acyclic Boolean conjunctive query. If each cycle in the attack graph of  $q$  is terminal, then each cycle in the attack graph has length 2.*

**Lemma 7** *Let  $q$  be an acyclic Boolean conjunctive query such that each cycle of the attack graph of  $q$  is terminal and each atom of  $q$  belongs to a cycle of the attack graph.*

1. If the same variable  $x$  occurs in two distinct cycles of the attack graph, then for each atom  $F$  in these cycles,  $x \in \text{key}(F)$ .
2. If  $F \stackrel{q}{\rightsquigarrow} G$  is a weak attack, then  $\text{key}(G) \subseteq \text{vars}(F)$ .

The following lemma applies to queries with an atom whose primary key contains no variables.

**Lemma 8** *Let  $q$  be a Boolean conjunctive query without self-join. Let  $F \in q$  such that  $\text{key}(F) = \emptyset$ . Let  $q' = q \setminus \{F\}$ . Let  $\vec{y}$  be a sequence of distinct variables such that  $\text{vars}(\vec{y}) = \text{vars}(F)$ . Let  $\mathbf{db}$  be an uncertain database that is purified relative to  $q$ , and let  $D$  be the active domain of  $\mathbf{db}$ . Then the following are equivalent:*

1.  $\mathbf{db} \in \text{CERTAINTY}(q)$ .
2.  $\mathbf{db} \neq \emptyset$  and for all  $\vec{b} \in D^{|\vec{y}|}$ , if  $F_{[\vec{y} \mapsto \vec{b}]} \in \mathbf{db}$ , then  $\mathbf{db} \in \text{CERTAINTY}(q'_{[\vec{y} \mapsto \vec{b}]})$ .

The proof of Theorem 3 can now be given.

**Proof Theorem 3** Given uncertain database  $\mathbf{db}$ , we need to show that it can be decided in polynomial time (in the size of  $\mathbf{db}$ ) whether  $\mathbf{db} \in \text{CERTAINTY}(q)$ . Let  $D$  be the active domain of  $\mathbf{db}$ . By Lemma 1, we can assume that  $\mathbf{db}$  is purified relative to  $q$ .

The proof runs by induction on the length of  $q$ . For the base of the induction, we consider the case where the attack graph of  $q$  contains no unattacked atom (i.e., no atom has zero indegree).  $\text{CERTAINTY}(q)$  is obviously in **P** if  $q = \{\}$ . Assume next that  $q$  is nonempty.

Since all cycles of  $q$ 's attack graph are terminal and every atom has an incoming attack, every atom of  $q$  belongs to some cycle of the attack graph. By Lemma 6, the attack graph of  $q$  is a set of disjoint weak cycles  $F_1 \rightsquigarrow G_1 \rightsquigarrow F_1, \dots, F_\ell \rightsquigarrow G_\ell \rightsquigarrow F_\ell$  for some  $\ell \geq 1$ . For  $i \in \{1, \dots, \ell\}$ , let  $q_i = \{F_i, G_i\}$ , and let  $\vec{x}_i$  be a sequence of distinct variables that contains every variable  $x \in \text{vars}(q_i)$  such that for some  $j \neq i$ ,  $x \in \text{vars}(q_j)$ . By Lemma 7,  $\text{vars}(x_i) \subseteq \text{key}(F_i) \cap \text{key}(G_i)$ .

For  $i \in \{1, \dots, \ell\}$ , let  $\mathbf{db}_i$  be the subset of  $\mathbf{db}$  containing every fact  $A$  with the same relation name as  $F_i$  or  $G_i$ . Call a *partition* of  $\mathbf{db}_i$  a maximal subset  $P$  of  $\mathbf{db}_i$  such that for some  $\vec{a} \in D^{|\vec{x}_i|}$ , for all  $A \in P$ , there exists a valuation  $\theta$  such that  $A = \theta_{[\vec{x}_i \mapsto \vec{a}]}(F_i)$  or  $A = \theta_{[\vec{x}_i \mapsto \vec{a}]}(G_i)$ . The sequence  $\vec{a}$  is called the *vector* of partition  $P$ .

In words, each partition of  $\mathbf{db}_i$  groups facts that can be obtained from  $F_i$  or  $G_i$  by replacing the variables of  $\vec{x}_i$  with the same fixed constants. For example, the attack graph in Fig. 4 contains an attack cycle involving  $R_3(\underline{x}, y, u_3, u_4)$  and  $R_4(x, y, u_4, u_3)$ . The sequence  $\langle x, y \rangle$  contains the variables that also occur in other cycles. The facts  $R_3(a, b, c, d)$  and  $R_4(a, b, e, f)$  both belong to the partition with vector  $\langle a, b \rangle$ .

Clearly, two facts that belong to distinct partitions of  $\mathbf{db}_i$  cannot be key-equal. It follows that each repair of  $\mathbf{db}_i$  is a disjoint union of repairs, one for each partition of  $\mathbf{db}_i$ .

Let  $\|\mathbf{db}_i\|$  be the smallest subset of  $\mathbf{db}_i$  that contains every partition  $P$  satisfying  $P \in \text{CERTAINTY}(q_i)$ . By

Lemma 7 and [13, Theorem 2],  $\text{CERTAINTY}(q_i)$  is in **P** for  $1 \leq i \leq \ell$ . From the following sublemma, it follows that  $\text{CERTAINTY}(q)$  is in **P**.

**Sublemma 5** *The following are equivalent:*

1.  $\mathbf{db} \in \text{CERTAINTY}(q)$ .
2.  $\bigcup_{1 \leq i \leq \ell} \|\mathbf{db}_i\| \models q$ .

For the step of the induction, assume that  $F$  is an unattacked atom in  $q$ 's attack graph. Let  $\vec{x}$  be a sequence of distinct variables such that  $\text{vars}(\vec{x}) = \text{key}(F)$ . By Corollary 8.11 in [23], the following are equivalent.

1.  $\mathbf{db} \in \text{CERTAINTY}(q)$ .
2. For some  $\vec{a} \in D^{|\vec{x}|}$ ,  $\mathbf{db} \in \text{CERTAINTY}(q_{[\vec{x} \mapsto \vec{a}]})$ .

Let  $\vec{y}$  be a sequence of distinct variables such that  $\text{vars}(\vec{y}) = \text{vars}(F) \setminus \text{key}(F)$ . Let  $q' = q \setminus \{F\}$ . By Lemma 1, it is possible to compute in polynomial time a database  $\mathbf{db}'$  that is purified relative to  $q_{[\vec{x} \mapsto \vec{a}]}$  such that

$$\mathbf{db} \in \text{CERTAINTY}(q_{[\vec{x} \mapsto \vec{a}]}) \iff \mathbf{db}' \in \text{CERTAINTY}(q_{[\vec{x} \mapsto \vec{a}]}).$$

By Lemma 8, the following are equivalent:

1.  $\mathbf{db}' \in \text{CERTAINTY}(q_{[\vec{x} \mapsto \vec{a}]})$ .
2.  $\mathbf{db}' \neq \emptyset$  and for all  $\vec{b} \in D^{|\vec{y}|}$ , if  $F_{[\vec{xy} \mapsto \vec{ab}]} \in \mathbf{db}'$ , then  $\mathbf{db}' \in \text{CERTAINTY}(q'_{[\vec{xy} \mapsto \vec{ab}]})$ .

By Lemma 5, all cycles in the attack graph of  $q'_{[\vec{xy} \mapsto \vec{ab}]}$  are weak and terminal. By the induction hypothesis, it follows that  $\text{CERTAINTY}(q'_{[\vec{xy} \mapsto \vec{ab}]})$  is in **P**. Since the sizes of  $D^{|\vec{x}|}$  and  $D^{|\vec{y}|}$  are polynomially bounded in the size of  $\mathbf{db}$ , it is correct to conclude that  $\text{CERTAINTY}(q)$  is in **P**.  $\square$

## 6.2 Nonterminal Weak Cycles

Theorems 2 and 3 leave open the complexity of  $\text{CERTAINTY}(q)$  when the attack graph of  $q$  contains one or more nonterminal weak cycles and no strong cycle. In this section, we zoom in on acyclic queries  $\text{AC}(k)$ , defined next for  $k \in \{2, 3, \dots\}$ , whose attack graph contains  $\frac{k(k-1)}{2}$  nonterminal weak cycles and no strong cycle. By showing tractability of  $\text{CERTAINTY}(\text{AC}(k))$ , we obtain more supporting evidence for Conjecture 1. As a side result, we will solve a complexity issue raised by Fuxman and Miller [10].

**Definition 8** For  $k \geq 2$ , let  $\text{C}(k)$  and  $\text{AC}(k)$  denote the following Boolean conjunctive queries without self-join.

$$\begin{aligned} \text{C}(k) &= \{R_1(\underline{x}_1, x_2), R_2(\underline{x}_2, x_3), \dots, R_{k-1}(\underline{x}_{k-1}, x_k), \\ &\quad R_k(x_k, x_1)\}, \\ \text{AC}(k) &= \{R_1(\underline{x}_1, x_2), R_2(\underline{x}_2, x_3), \dots, R_{k-1}(\underline{x}_{k-1}, x_k), \\ &\quad R_k(x_k, x_1), S_k(x_1, x_2, \dots, x_k)\}, \end{aligned}$$

where  $x_1, \dots, x_k$  are distinct variables and  $R_1, \dots, R_k, S_k$  distinct relation names. For  $i \in \{1, \dots, k\}$ , relation name  $R_i$  is of signature  $[2, 1]$ , and  $S_k$  is of signature  $[k, k]$ .  $\triangleleft$



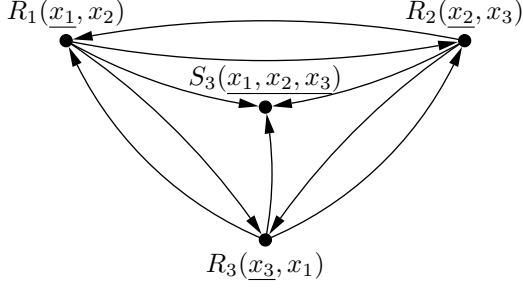


Figure 5: Attack graph of AC(3). All cycles are weak and nonterminal

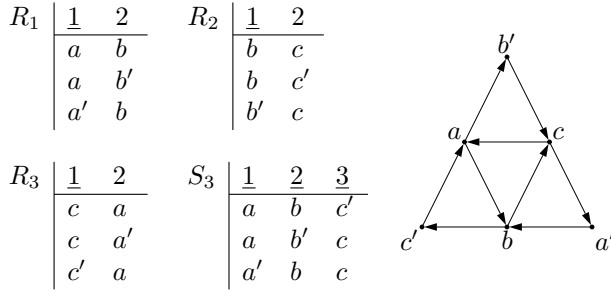


Figure 6: At the left: uncertain database that is purified relative to AC(3). At the right: graph representation of  $R_1 \cup R_2 \cup R_3$ . Note that the three cycles encoded in  $S_3$  are clockwise.

Obviously, a query  $q$  is acyclic if it contains an atom  $F$  such that  $\text{vars}(F) = \text{vars}(q)$ . Therefore,  $\text{AC}(k)$  is acyclic because the  $S_k$ -atom contains all variables that occur in the query. On the other hand,  $\text{C}(k)$  is acyclic if  $k = 2$  and cyclic if  $k \geq 3$ .

For  $i \in \{1, \dots, k\}$ , the attack graph of  $\text{AC}(k)$  contains attacks from the  $R_i$ -atom to every other atom. Figure 5 shows the attack graph of  $\text{AC}(3)$ . All attack cycles are weak, but Theorem 3 does not apply because the cycles are nonterminal.

$\text{CERTAINTY}(\text{C}(k))$  was claimed **coNP**-hard for all  $k \geq 2$  in [10]. Later, however, Wijzen [22] found a mistake in the proof of that claim and showed that  $\text{CERTAINTY}(\text{C}(k))$  is tractable if  $k = 2$ . The complexity of  $\text{CERTAINTY}(\text{C}(k))$  for  $k \geq 3$  will be settled by Corollary 1.

**Theorem 4** For  $k \geq 2$ ,  $\text{CERTAINTY}(\text{AC}(k))$  is in **P**.

**Proof (Extended sketch.)** Let  $\mathbf{db}$  be an uncertain database with schema  $\{R_1, \dots, R_k, S_k\}$ . By Lemma 1, we can assume without loss of generality that  $\mathbf{db}$  is purified relative to  $\text{AC}(k)$ . Let  $D$  be the active domain of  $\mathbf{db}$ . For every  $i \in \{1, \dots, k\}$ , define  $\text{type}(x_i)$  as the subset of  $D$  that contains  $a$  if for some valuation  $\mu$ ,  $\mu_{[x_i \mapsto a]}(\text{AC}(k)) \subseteq \mathbf{db}$ . Since  $\text{AC}(k)$  has no self-join, we can assume without loss of generality that  $i \neq j$  implies  $\text{type}(x_i) \cap \text{type}(x_j) = \emptyset$ .

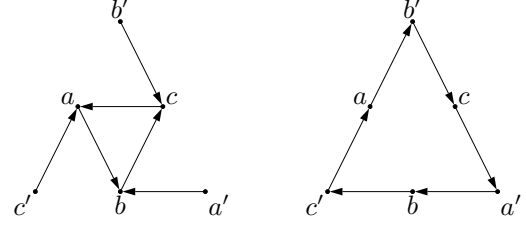


Figure 7: Graph representation of two repairs (of the uncertain database of Fig. 6) that falsify  $\text{AC}(3)$ . The left cycle is anticlockwise and not encoded in  $S_3$ .

For example, assume  $R_i(\underline{a}, b), R_j(\underline{c}, d) \in \mathbf{db}$  with  $i < j$ . Since  $a \in \text{type}(x_i)$ ,  $b \in \text{type}(x_{i+1})$ , and  $c \in \text{type}(x_j)$ , it follows that  $b \neq a \neq c$  and that  $b = c$  implies  $j = i + 1$ .

The  $R_i$ -facts of  $\mathbf{db}$  can be viewed as edges of a directed graph ( $1 \leq i \leq k$ ). This is illustrated in Fig. 6 for  $k = 3$ . Let  $G = (V, E)$  be the directed graph such that  $V = D$  and  $E = \{(a, b) \mid R_i(\underline{a}, b) \in \mathbf{db} \text{ for some } i\}$ . Then,  $G$  is  $k$ -partite with vertex classes  $\text{type}(x_1), \dots, \text{type}(x_k)$ . Furthermore, whenever  $(a, b) \in E$  and  $a \in \text{type}(x_i)$ , then  $b \in \text{type}(x_{i+1})$  if  $i < k$  (and  $b \in \text{type}(x_1)$  if  $i = k$ ). It follows that the length of every elementary cycle in  $G$  must be a multiple of  $k$ . Since  $\mathbf{db}$  is purified, no vertex has zero outdegree. We define  $\mathcal{C}$  as the set of cycles of length  $k$  such that if  $\mathbf{db}$  contains  $S_k(\underline{a_1}, \dots, \underline{a_k})$ , then  $\mathcal{C}$  contains the cycle  $a_1, a_2, \dots, a_k, a_1$ .

Since  $\mathbf{db}$  is purified,  $G$  is a vertex-disjoint union of strong components  $S_1, \dots, S_\ell$  (for some  $\ell \geq 0$ ) such that for  $i \neq j$ , no edge leads from a vertex in  $S_i$  to a vertex in  $S_j$ .<sup>2</sup>

In what follows, some vertices and edges of  $G$  will be *marked*. It is straightforward that  $\mathbf{db} \notin \text{CERTAINTY}(\text{AC}(k))$  is equivalent to the following.

It is possible to mark exactly one outgoing edge for each vertex of  $G$  without marking all edges of some cycle in  $\mathcal{C}$ . (5)

We provide a polynomial-time algorithm for testing condition (5). Marking one outgoing edge for each vertex will create a cycle of marked edges in each strong component.

For each strong component  $S_i$ , consider the following cases successively and execute the first one that applies.

**Case  $S_i$  contains a cycle of length  $k$  that does not belong to  $\mathcal{C}$ .** Such a cycle is illustrated by Fig. 7 (left). Mark all vertices and edges of the cycle. Notice that the number of cycles of length  $k$  is at most  $|V|^k$ , which is polynomial in the size of  $\mathbf{db}$ .

**Case  $S_i$  contains an elementary cycle of length (strictly) greater than  $k$ .** Such a cycle is illustrated by Fig. 7 (right). Mark all vertices and edges of the cycle. To see that this step is in polynomial time, notice that the following are equivalent:

- $S_i$  contains an elementary cycle of length greater than  $k$ .

<sup>2</sup>A strong component of a graph  $G$  is a maximal strongly connected subgraph of  $G$ . A graph is strongly connected if there is a path from any vertex to any other.

- $S_i$  contains a path  $a_1, a_2, \dots, a_k, a_{k+1}$  such that  $a_1 \neq a_{k+1}$  and  $S_i$  contains a path from  $a_{k+1}$  to  $a_1$  that contains no edge from  $\{a_1, a_2, \dots, a_k\} \times V$ .

The latter condition can be tested in polynomial time, because there are at most  $|V|^{k+1}$  distinct choices for  $a_1, a_2, \dots, a_k, a_{k+1}$  and paths can be found in polynomial time.

**Case neither of the above two cases applies.** Conclude that (5) is false.

If after the previous step every strong component contains a cycle of marked edges, then it is correct to conclude that (5) is true. Notice that every cycle of  $\mathcal{C}$  now contains at least one unmarked edge. We can achieve (5) by marking, for each yet unmarked vertex, the vertices and edges on a shortest path to some marked vertex. This can be done without creating new cycles of marked edges.  $\square$

Since query  $C(k)$  is acyclic if  $k \geq 3$ , attack graphs are not defined for  $C(k)$  if  $k \geq 3$ . Nevertheless, the following lemma immediately implies that if  $\text{CERTAINTY}(\text{AC}(k))$  is tractable, then so is  $\text{CERTAINTY}(C(k))$ .

**Lemma 9** *Let  $q$  be a Boolean conjunctive query without self-join. If  $q' \subseteq q$  and every atom in  $q \setminus q'$  is all-key, then there exists an  $\text{AC}^0$  many-one reduction from  $\text{CERTAINTY}(q')$  to  $\text{CERTAINTY}(q)$ .*

**Corollary 1** *For  $k \geq 2$ ,  $\text{CERTAINTY}(C(k))$  is in  $\mathbf{P}$ .*

Unsurprisingly, there exist acyclic queries  $q \notin \{\text{AC}(k) \mid k \geq 2\}$  whose attack graph contains some nonterminal cycle and no strong cycle. The complexity of  $\text{CERTAINTY}(q)$  for such queries  $q$  is open.

## 7 Uncertainty and Probability

In this section, we study the relationship between the complexities of  $\text{CERTAINTY}(q)$  and evaluating  $q$  on probabilistic databases. The motivation is that, on input of an uncertain database  $\mathbf{db}$ , the problem  $\text{CERTAINTY}(q)$  is solved if we can determine whether query  $q$  evaluates to probability 1 on the probabilistic database obtained from  $\mathbf{db}$  by assuming a uniform probability distribution over the set of repairs of  $\mathbf{db}$ . We show, however, that this approach provides no new insights in the tractability frontier of  $\text{CERTAINTY}(q)$ .

### 7.1 Background from Probabilistic Databases

In this section, we review an important result from probabilistic database theory.

**Definition 9** A possible world  $\mathbf{w}$  of uncertain database  $\mathbf{db}$  is a consistent subset of  $\mathbf{db}$ . The set of possible worlds of  $\mathbf{db}$  is denoted  $\text{worlds}(\mathbf{db})$ . Notice that possible worlds, unlike repairs, need not be maximal consistent.

A probabilistic database is a pair  $(\mathbf{db}, \text{Pr})$  where  $\mathbf{db}$  is an uncertain database and  $\text{Pr} : \text{worlds}(\mathbf{db}) \rightarrow [0, 1]$  is a total function such that  $\sum_{\mathbf{w} \in \text{worlds}(\mathbf{db})} \text{Pr}(\mathbf{w}) = 1$ . We will assume that the numbers in the codomain of  $\text{Pr}$  are rational.  $\triangleleft$

The following definition extends the function  $\text{Pr}$  to Boolean first-order queries  $q$ .

**Definition 10** Let  $(\mathbf{db}, \text{Pr})$  be a probabilistic database. Let  $q$  be a Boolean first-order query. We define

$$\text{Pr}(q) = \sum_{\mathbf{w} \in \text{worlds}(\mathbf{db}) : \mathbf{w} \models q} \text{Pr}(\mathbf{w}).$$

In words,  $\text{Pr}(q)$  sums up the probabilities of the possible worlds that satisfy  $q$ .  $\triangleleft$

Of special interest is the application of Definition 10 in case  $q$  is a single fact, or a Boolean combination of facts. Notice that if  $(\mathbf{db}, \text{Pr})$  is a probabilistic database and  $A_1, \dots, A_n$  are distinct facts belonging to a same block of  $\mathbf{db}$ , then  $\text{Pr}(A_1 \vee A_2 \vee \dots \vee A_n) = \sum_{i=1}^n \text{Pr}(A_i)$ , because no possible world can contain two distinct facts that belong to a same block.

**Definition 11** Probabilistic database  $(\mathbf{db}, \text{Pr})$  is called *block-independent-disjoint* (BID) if the following holds: whenever  $A_1, \dots, A_n$  are facts of  $\mathbf{db}$  taken from  $n$  distinct blocks (for some  $n \geq 1$ ), then  $\text{Pr}(A_1 \wedge A_2 \wedge \dots \wedge A_n) = \prod_{i=1}^n \text{Pr}(A_i)$ .  $\triangleleft$

Theorem 2.4 in [8] implies that every BID probabilistic database  $(\mathbf{db}, \text{Pr})$  is uniquely determined if  $\text{Pr}(A)$  is given for every fact  $A \in \mathbf{db}$ . This allows for an efficient encoding: rather than specifying  $\text{Pr}(\mathbf{w})$  for every  $\mathbf{w} \in \text{worlds}(\mathbf{db})$ , it suffices to specify  $\text{Pr}(A)$  for every  $A \in \mathbf{db}$ . In the complexity results that follow, this efficient encoding is assumed.

Notice that we can turn an uncertain database  $\mathbf{db}$  into a BID probabilistic database by assuming that the probabilities of all repairs are equal and sum up to 1. A consistent subset of  $\mathbf{db}$  that is not maximal, would then have zero probability.

---

**Function**  $\text{IsSafe}(q)$  Determine whether  $q$  is safe [8]

---

**Input:**  $q$  is a Boolean conjunctive query without self-join.

**Result:** Boolean in  $\{\text{true}, \text{false}\}$ .

**begin**

R1: **if**  $|q| = 1$  and  $\text{vars}(q) = \emptyset$  **then**

**return true;**

R2: **if**  $q = q_1 \cup q_2$  with  $q_1 \neq \emptyset \neq q_2$ ,  $\text{vars}(q_1) \cap \text{vars}(q_2) = \emptyset$  **then**

**return**  $\text{IsSafe}(q_1) \wedge \text{IsSafe}(q_2)$ ;

/\*  $a$  is an arbitrary constant \*/

R3: **if**  $\bigcap_{F \in q} \text{key}(F) \neq \emptyset$  **then**

    select  $x \in \bigcap_{F \in q} \text{key}(F)$ ;

**return**  $\text{IsSafe}(q_{[x \rightarrow a]})$ ;

R4: **if** there exists  $F \in q$  such that  $\text{key}(F) = \emptyset \neq \text{vars}(F)$  **then**

    select  $F \in q$  such that  $\text{key}(F) = \emptyset \neq \text{vars}(F)$ ; select  $x \in \text{vars}(F)$ ;

**return**  $\text{IsSafe}(q_{[x \rightarrow a]})$ ;

**if none of the above then**

**return false;**

---

**Definition 12** Given a Boolean query  $q$ ,  $\text{PROBABILITY}(q)$  is the following function problem: on input of a BID probabilistic database  $(\mathbf{db}, \text{Pr})$ , determine the value of  $\text{Pr}(q)$ .

A Boolean conjunctive query  $q$ , without self-join, is called *safe* if Algorithm IsSafe returns **true**.  $\triangleleft$

The following result establishes a dichotomy in the complexity of  $\text{PROBABILITY}(q)$ .

**Theorem 5 ([8])** *Let  $q$  be a Boolean conjunctive query without self-join.*

1. *If  $q$  is safe, then  $\text{PROBABILITY}(q)$  is in **FP**.*
2. *If  $q$  is not safe, then  $\text{PROBABILITY}(q)$  is  $\sharp\mathbf{P}$ -hard.*

## 7.2 Comparing Complexities

The following proposition establishes a straightforward relationship between the problems  $\text{PROBABILITY}(q)$  and  $\text{CERTAINTY}(q)$ . The only subtlety is that a repair contains a fact of each block, while a possible world and a block may have an empty intersection (recall that possible worlds, unlike repairs, need not be maximal consistent). In the statement of this proposition,  $\mathbf{db}'$  restricts  $\mathbf{db}$  to the set of blocks whose probabilities sum up to 1.

**Proposition 1** *Let  $(\mathbf{db}, \text{Pr})$  be a BID probabilistic database. Let  $\mathbf{db}'$  be the smallest subset of  $\mathbf{db}$  that contains every block  $\mathbf{b}$  of  $\mathbf{db}$  such that  $\sum_{A \in \mathbf{b}} \text{Pr}(A) = 1$ . Let  $q$  be a Boolean conjunctive query. Then the following are equivalent:*

1.  $\mathbf{db}' \in \text{CERTAINTY}(q)$ .
2. *On input  $(\mathbf{db}, \text{Pr})$ , the answer to the function problem  $\text{PROBABILITY}(q)$  is 1.*

The following theorem establishes a nontrivial relationship between the complexities of  $\text{CERTAINTY}(q)$  and  $\text{PROBABILITY}(q)$ . Notice that the query  $q$  in the theorem's statement is not required to be acyclic.

**Theorem 6** *Let  $q$  be a Boolean conjunctive query without self-join. If  $q$  is safe, then  $\text{CERTAINTY}(q)$  is first-order expressible.*

**Proof** The proof runs by induction on the execution of Algorithm IsSafe. Since  $q$  is safe, some rule of IsSafe applies to  $q$ .

**Case R1 applies.** If  $q$  consists of a single fact, then  $\text{CERTAINTY}(q)$  is obviously first-order expressible.

**Case R2 applies.** Let  $q = q_1 \cup q_2$  with  $q_1 \neq \emptyset \neq q_2$  and  $\text{vars}(q_1) \cap \text{vars}(q_2) = \emptyset$ . Since  $q$  is safe,  $q_1$  and  $q_2$  are safe by definition of safety. By the induction hypothesis, there exists a certain first-order rewriting  $\phi_1$  of  $q_1$ , and a certain first-order rewriting  $\phi_2$  of  $q_2$ . Obviously,  $\phi_1 \wedge \phi_2$  is a certain first-order rewriting of  $q$ .

**Case R3 applies.** Assume variable  $x$  such that for every  $F \in q$ ,  $x \in \text{key}(F)$ . By definition of safety,  $q_{[x \rightarrow a]}$  is safe. It

can be easily seen that  $\mathbf{db} \in \text{CERTAINTY}(q)$  if and only if for some constant  $a$ ,  $\mathbf{db} \in \text{CERTAINTY}(q_{[x \rightarrow a]})$ . By the induction hypothesis,  $\text{CERTAINTY}(q_{[x \rightarrow a]})$  is first-order expressible. Let  $\phi$  be a certain first-order rewriting of  $q_{[x \rightarrow c]}$ , where we assume without loss of generality that  $c$  is a constant that does not occur in  $q$ . Let  $\phi(x)$  be the first order formula obtained from  $\phi$  by replacing each occurrence of  $c$  with  $x$ . Then,  $\exists x \phi(x)$  of a certain first-order rewriting of  $q$ .

**Case R4 applies.** Assume  $F \in q$  such that  $\text{key}(F) = \emptyset$  and  $\text{vars}(F) \neq \emptyset$ . Let  $\vec{x}$  be a sequence of distinct variables such that  $\text{vars}(\vec{x}) = \text{vars}(F)$ . Let  $\vec{a} = \langle a, a, \dots, a \rangle$  be a sequence of length  $|\vec{x}|$ . By definition of safety,  $q_{[\vec{x} \rightarrow \vec{a}]}$  is safe. By the induction hypothesis,  $\text{CERTAINTY}(q_{[\vec{x} \rightarrow \vec{a}]})$  is first-order expressible. From Lemma 8.6 in [23], it follows that  $\text{CERTAINTY}(q)$  is first-order expressible  $\square$

**Corollary 2** *Let  $q$  be a Boolean conjunctive query without self-join. If  $\text{CERTAINTY}(q)$  is not first-order expressible, then the function problem  $\text{PROBABILITY}(q)$  is  $\sharp\mathbf{P}$ -hard.*

For acyclic queries, the only complexities of  $\text{CERTAINTY}(q)$  left open by Theorems 1, 2, and 3 concern queries  $q$  with a cyclic attack graph (in particular, an attack graph without strong cycle and with at least one nonterminal weak cycle). For such a query  $q$ ,  $\text{CERTAINTY}(q)$  is not first-order expressible (by Theorem 1), hence  $\text{PROBABILITY}(q)$  is  $\sharp\mathbf{P}$ -hard (by Corollary 2). Consequently, the probabilistic database approach fails to provide further insight into the tractability frontier of  $\text{CERTAINTY}(q)$ . It turns out that the queries  $q$  for which  $\text{PROBABILITY}(q)$  is tractable is a very restricted subset of the queries for which  $\text{CERTAINTY}(q)$  is tractable (assuming  $\mathbf{FP} \neq \sharp\mathbf{P}$  and  $\mathbf{P} \neq \mathbf{coNP}$ ).

## 8 Discussion

In the following, we say that a class  $\mathcal{P}$  of function problems *exhibits an effective  $\mathbf{FP}$ - $\sharp\mathbf{P}$ -dichotomy* if all problems in  $\mathcal{P}$  are either in  $\mathbf{FP}$  or  $\sharp\mathbf{P}$ -hard and it is decidable whether a given problem in  $\mathcal{P}$  is in  $\mathbf{FP}$  or  $\sharp\mathbf{P}$ -hard. Likewise, we say that a class  $\mathcal{P}$  of decision problems *exhibits an effective  $\mathbf{P}$ - $\mathbf{coNP}$ -dichotomy* if all problems in  $\mathcal{P}$  are either in  $\mathbf{P}$  or  $\mathbf{coNP}$ -hard and it is decidable whether a given problem in  $\mathcal{P}$  is in  $\mathbf{P}$  or  $\mathbf{coNP}$ -hard.

Recall from Section 2 that  $\sharp\text{CERTAINTY}(q)$  is the counting variant of  $\text{CERTAINTY}(q)$ , which takes as input an uncertain database  $\mathbf{db}$  and asks how many repairs of  $\mathbf{db}$  satisfy query  $q$ . For the probabilistic and counting variants of  $\text{CERTAINTY}(q)$ , the following dichotomies have been established.

**Theorem 7 ([8],[15])** *The following classes exhibit an effective  $\mathbf{FP}$ - $\sharp\mathbf{P}$ -dichotomy:*

1. *the class containing  $\text{PROBABILITY}(q)$  for all Boolean conjunctive queries  $q$  without self-join; and*
2. *the class containing  $\sharp\text{CERTAINTY}(q)$  for all Boolean conjunctive queries  $q$  without self-join.*

Theorem 2 and Conjecture 1 imply the following conjecture, which is thus weaker than Conjecture 1.

**Conjecture 2** *The class containing  $\text{CERTAINTY}(q)$  for all acyclic Boolean conjunctive queries  $q$  without self-join exhibits an effective  $\mathbf{P}$ -coNP-dichotomy.*

From Theorems 2 and 3, it follows that in order to prove Conjecture 2, it suffices to show that an effective  $\mathbf{P}$ -coNP-dichotomy is exhibited by the class containing  $\text{CERTAINTY}(q)$  for all queries  $q$  whose attack graph contains some nonterminal cycle and no strong cycle.

We confidently believe that the  $\mathbf{P}$ -coNP-dichotomy of Conjecture 2 (if true) will be harder to prove than the  $\mathbf{FP}$ - $\mathbf{P}$ -dichotomies established by Theorem 7, for the following reasons. All problems  $\text{PROBABILITY}(q)$  that are in  $\mathbf{FP}$  can be solved by a single, fairly simple polynomial-time algorithm which appears in [8]. Likewise, all problems  $\mathbf{P}$ - $\text{CERTAINTY}(q)$  in  $\mathbf{FP}$  can be solved by a single, fairly simple polynomial-time algorithm [15]. On the other hand,  $\text{CERTAINTY}(q)$  problems in  $\mathbf{P}$  seem to ask for sophisticated polynomial-time algorithms. In their proof that Conjecture 2 holds for queries with exactly two atoms, Kolaitis and Pema [13] made use of an ingenious polynomial-time algorithm of Minty [17]. Our proof of Theorem 4 uses algorithms from (directed) graph theory. Despite their sophistication, these polynomial-time algorithms only solve restricted cases of  $\text{CERTAINTY}(q)$ .

Notice also that by Corollary 2 and Theorem 1, the function problem  $\text{PROBABILITY}(q)$  is intractable for all acyclic queries  $q$  with a cyclic attack graph. On the other hand, cycles in attack graphs are exactly what makes Conjecture 2 hard to prove.

## References

- [1] ABITEBOUL, S., HULL, R., AND VIANU, V. *Foundations of Databases*. Addison-Wesley, 1995.
- [2] ARENAS, M., BERTOSSI, L. E., AND CHOMICKI, J. Consistent query answers in inconsistent databases. In *PODS* (1999), ACM Press, pp. 68–79.
- [3] BEERI, C., FAGIN, R., MAIER, D., AND YANNAKAKIS, M. On the desirability of acyclic database schemes. *J. ACM* 30, 3 (1983), 479–513.
- [4] BERTOSSI, L. E. *Database Repairing and Consistent Query Answering*. Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2011.
- [5] BIENVENU, M. Inconsistency-tolerant conjunctive query answering for simple ontologies. In *Description Logics* (2012), Y. Kazakov, D. Lembo, and F. Wolter, Eds., vol. 846 of *CEUR Workshop Proceedings*, CEUR-WS.org.
- [6] CHOMICKI, J., AND MARCINKOWSKI, J. Minimal-change integrity maintenance using tuple deletions. *Inf. Comput.* 197, 1-2 (2005), 90–121.
- [7] DALVI, N. N., RÉ, C., AND SUCIU, D. Probabilistic databases: diamonds in the dirt. *Commun. ACM* 52, 7 (2009), 86–94.
- [8] DALVI, N. N., RE, C., AND SUCIU, D. Queries and materialized views on probabilistic databases. *J. Comput. Syst. Sci.* 77, 3 (2011), 473–490.
- [9] FUXMAN, A., AND MILLER, R. J. First-order query rewriting for inconsistent databases. In *ICDT* (2005), T. Eiter and L. Libkin, Eds., vol. 3363 of *Lecture Notes in Computer Science*, Springer, pp. 337–351.
- [10] FUXMAN, A., AND MILLER, R. J. First-order query rewriting for inconsistent databases. *J. Comput. Syst. Sci.* 73, 4 (2007), 610–635.
- [11] GOTTLÖB, G., LEONE, N., AND SCARCELLO, F. Hypertree decompositions and tractable queries. *J. Comput. Syst. Sci.* 64, 3 (2002), 579–627.
- [12] GRIECO, L., LEMBO, D., ROSATI, R., AND RUZZI, M. Consistent query answering under key and exclusion dependencies: algorithms and experiments. In *CIKM* (2005), O. Herzog, H.-J. Schek, N. Fuhr, A. Chowdhury, and W. Teiken, Eds., ACM, pp. 792–799.
- [13] KOLAITIS, P. G., AND PEMA, E. A dichotomy in the complexity of consistent query answering for queries with two atoms. *Inf. Process. Lett.* 112, 3 (2012), 77–85.
- [14] LADNER, R. E. On the structure of polynomial time reducibility. *J. ACM* 22, 1 (1975), 155–171.
- [15] MASLOWSKI, D., AND WIJSEN, J. A dichotomy in the complexity of counting database repairs. *J. Comput. Syst. Sci.*. In press.
- [16] MASLOWSKI, D., AND WIJSEN, J. On counting database repairs. In *LID* (2011), G. H. L. Fletcher and S. Staworko, Eds., ACM, pp. 15–22.
- [17] MINTY, G. J. On maximal independent sets of vertices in claw-free graphs. *J. Comb. Theory, Ser. B* 28, 3 (1980), 284–304.
- [18] SUCIU, D., OLTEANU, D., RÉ, C., AND KOCH, C. *Probabilistic Databases*. Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2011.
- [19] ULLMAN, J. D. *Principles of Database and Knowledge-Base Systems, Volume I*. Computer Science Press, 1988.
- [20] WIJSEN, J. On the consistent rewriting of conjunctive queries under primary key constraints. *Inf. Syst.* 34, 7 (2009), 578–601.
- [21] WIJSEN, J. On the first-order expressibility of computing certain answers to conjunctive queries over uncertain databases. In *PODS* (2010), J. Paredaens and D. V. Gucht, Eds., ACM, pp. 179–190.
- [22] WIJSEN, J. A remark on the complexity of consistent conjunctive query answering under primary key violations. *Information Processing Letters* 110, 21 (2010), 950 – 955.
- [23] WIJSEN, J. Certain conjunctive query answering in first-order logic. *ACM Trans. Database Syst.* 37, 2 (2012), 9.

## A Proofs of Section 3

**Proof Lemma 1** In polynomial time, we can construct a maximal sequence  $\mathbf{db}_0 \xrightarrow{A_1} \mathbf{db}_1 \xrightarrow{A_2} \mathbf{db}_2 \cdots \xrightarrow{A_n} \mathbf{db}_n$  such that for  $i \in \{1, 2, \dots, n\}$ ,

1.  $A_i \in \mathbf{db}_{i-1}$ ;
2. there exists no valuation  $\theta$  such that  $A_i \in \theta(q) \subseteq \mathbf{db}_{i-1}$ ; and
3.  $\mathbf{db}_i = \mathbf{db}_{i-1} \setminus \text{block}(A_i, \mathbf{db}_{i-1})$ .

Clearly,  $\mathbf{db}_n$  is purified relative to  $q$ . We show that for  $i \in \{1, 2, \dots, n\}$ ,

$$\mathbf{db}_{i-1} \in \text{CERTAINTY}(q) \iff \mathbf{db}_i \in \text{CERTAINTY}(q).$$

$\implies$  By contraposition. Assume that  $\mathbf{r}$  is a repair of  $\mathbf{db}_i$  such that  $\mathbf{r} \not\models q$ . Then,  $\mathbf{r} \cup \{A_i\}$  is a repair of  $\mathbf{db}_{i-1}$  that falsifies  $q$ .  $\impliedby$  By contraposition. Assume that  $\mathbf{r}$  is a repair of  $\mathbf{db}_{i-1}$  such that  $\mathbf{r} \not\models q$ . Obviously,  $\mathbf{r} \setminus \text{block}(A_i, \mathbf{db}_{i-1})$  is a repair of  $\mathbf{db}_i$  that falsifies  $q$ .  $\square$

## B Proofs of Section 4

**Proof Lemma 2** Assume  $F \rightsquigarrow G$ . Let  $\tau$  be a join tree for  $q$ . Let  $F \xrightarrow{L_1} F_1 \cdots \xrightarrow{L_{m-1}} F_{m-1} \xrightarrow{L_m} G$  be the path in  $\tau$  between  $F$  and  $G$  ( $m \geq 1$ ).

We have  $L_m \subseteq \text{vars}(G)$ . Since  $\mathcal{K}(q \setminus \{F\}) \models \text{key}(G) \rightarrow L_m$  and  $L_m \not\subseteq F^{+,q}$  (because  $F \rightsquigarrow G$ ), it must be the case that  $\mathcal{K}(q \setminus \{F\}) \not\models \text{key}(F) \rightarrow \text{key}(G)$ , hence  $\text{key}(G) \not\subseteq F^{+,q}$ .

We have  $L_1 \subseteq \text{vars}(F)$ . Since  $L_1 \not\subseteq F^{+,q}$  (because  $F \rightsquigarrow G$ ), it must be the case that  $\text{vars}(F) \not\subseteq F^{+,q}$ .  $\square$

## C Proofs of Section 5

**Proof Lemma 4** We show that if the attack graph of  $q$  contains a strong cycle of length  $n$  with  $n \geq 3$ , then it contains a strong cycle of some length  $m$  with  $m < n$ .

Let  $H_0 \rightsquigarrow H_1 \rightsquigarrow H_2 \cdots \rightsquigarrow H_{n-1} \rightsquigarrow H_0$  be a strong cycle of length  $n$  ( $n \geq 3$ ) in the attack graph of  $q$ , where  $i \neq j$  implies  $H_i \neq H_j$ . Assume without loss of generality that the attack  $H_0 \rightsquigarrow H_1$  is strong. Thus,  $\mathcal{K}(q) \not\models \text{key}(H_0) \rightarrow \text{key}(H_1)$ .

We write  $i \oplus j$  as shorthand for  $(i + j) \bmod n$ . If  $H_1 \rightsquigarrow H_{1 \oplus 2}$ , then  $H_0 \rightsquigarrow H_1 \rightsquigarrow H_{1 \oplus 2} \cdots \rightsquigarrow H_{n-1} \rightsquigarrow H_0$  is a strong cycle of length  $n - 1$ , and the desired result holds. Assume next  $H_1 \not\rightsquigarrow H_{1 \oplus 2}$ . By Lemma 3,  $H_2 \rightsquigarrow H_1$ . We distinguish two cases.

**Case  $H_2 \rightsquigarrow H_1$  is a strong attack.** Then  $H_1 \rightsquigarrow H_2 \rightsquigarrow H_1$  is a strong cycle of length  $2 < n$ .

**Case  $H_2 \rightsquigarrow H_1$  is a weak attack.** If  $H_1 \rightsquigarrow H_0$ , then  $H_0 \rightsquigarrow H_1 \rightsquigarrow H_0$  is a strong cycle of length  $2 < n$ . Assume next  $H_1 \not\rightsquigarrow H_0$ . Then, from  $H_0 \rightsquigarrow H_1 \rightsquigarrow H_2$  and Lemma 3, it follows  $H_0 \rightsquigarrow H_2$ . The cycle  $H_0 \rightsquigarrow H_2 \rightsquigarrow H_{2 \oplus 1} \cdots \rightsquigarrow H_{n-1} \rightsquigarrow H_0$  has length  $n - 1$ . It suffices to show that the attack  $H_0 \rightsquigarrow H_2$  is strong. Assume towards a contradiction that the attack  $H_0 \rightsquigarrow H_2$  is weak. Then,  $\mathcal{K}(q) \models \text{key}(H_0) \rightarrow \text{key}(H_2)$ . Since

$H_2 \rightsquigarrow H_1$  is a weak attack,  $\mathcal{K}(q) \models \text{key}(H_2) \rightarrow \text{key}(H_1)$ . By transitivity,  $\mathcal{K}(q) \models \text{key}(H_0) \rightarrow \text{key}(H_1)$ , a contradiction. This concludes the proof.  $\square$

**Proof Sublemma 2**  $\boxed{1. \implies}$  Obviously,  $\text{key}(F) \subseteq F^{+,q}$ . From  $G \rightsquigarrow F$  and Lemma 2, it follows  $\text{key}(F) \not\subseteq G^{+,q}$ . Thus, we can assume  $u \in \text{key}(F)$  such that  $u \in F^{+,q} \setminus G^{+,q}$ . Hence,  $\hat{\theta}_1(u) = \theta_1(x)$  and  $\hat{\theta}_2(u) = \theta_2(x)$ . Since  $\hat{\theta}_1(F)$  and  $\hat{\theta}_2(F)$  are key-equal by the premise,  $\hat{\theta}_1$  and  $\hat{\theta}_2$  agree on all variables of  $\text{key}(F)$ . In particular,  $\hat{\theta}_1(u) = \hat{\theta}_2(u)$ . It follows  $\theta_1(x) = \theta_2(x)$ .  $\boxed{1. \impliedby}$  Assume  $\theta_1(x) = \theta_2(x)$ . Since  $\text{key}(F) \subseteq F^{+,q}$ , and since neither  $y$  nor  $z$  occurs inside  $F^{+,q}$  in the Venn diagram of Fig. 3, it is correct to conclude that  $\hat{\theta}_1(F)$  and  $\hat{\theta}_2(F)$  are key-equal.

$\boxed{2. \implies}$  From  $F \rightsquigarrow G$  and Lemma 2, it follows  $\text{vars}(F) \not\subseteq F^{+,q}$ . Since  $\text{key}(F) \subseteq F^{+,q}$ , we can assume a variable  $u \in \text{vars}(F) \setminus \text{key}(F)$  such that  $u \notin F^{+,q}$ . From the premise  $\hat{\theta}_1(F) = \hat{\theta}_2(F)$ , it follows  $\hat{\theta}_1(u) = \hat{\theta}_2(u)$ . Since  $y$  occurs in all regions outside  $F^{+,q}$  in the Venn diagram, we conclude  $\theta_1(y) = \theta_2(y)$ . Finally,  $\theta_1(x) = \theta_2(x)$  follows from item 1 proved before.  $\boxed{2. \impliedby}$  Assume  $\theta_1(x) = \theta_2(x)$  and  $\theta_1(y) = \theta_2(y)$ . Since  $\text{vars}(F) \subseteq F^{\boxplus,q}$ , and since  $z$  does not occur inside  $F^{\boxplus,q}$  in the Venn diagram, it is correct to conclude  $\hat{\theta}_1(F) = \hat{\theta}_2(F)$ . This concludes the proof of Sublemma 2.  $\dashv$

**Proof Sublemma 3**  $\boxed{1. \implies}$  Obviously,  $\text{key}(G) \subseteq G^{+,q}$ . Since  $F \rightsquigarrow G$  is a strong attack,  $\text{key}(G) \not\subseteq F^{\boxplus,q}$ . We can assume  $u \in \text{key}(G)$  such that  $u \in G^{+,q} \setminus F^{\boxplus,q}$ . Consequently,  $\hat{\theta}_1(u) = \langle \theta_1(y), \theta_1(z) \rangle$  and  $\hat{\theta}_2(u) = \langle \theta_2(y), \theta_2(z) \rangle$ . Since  $\hat{\theta}_1(F)$  and  $\hat{\theta}_2(F)$  are key-equal by the premise,  $\hat{\theta}_1$  and  $\hat{\theta}_2$  agree on all variables of  $\text{key}(G)$ . In particular,  $\hat{\theta}_1(u) = \hat{\theta}_2(u)$ . It follows  $\theta_1(y) = \theta_2(y)$  and  $\theta_1(z) = \theta_2(z)$ .  $\boxed{1. \impliedby}$  Assume  $\theta_1(y) = \theta_2(y)$  and  $\theta_1(z) = \theta_2(z)$ . Since  $\text{key}(G) \subseteq G^{+,q}$ , and since  $x$  does not occur inside  $G^{+,q}$  in the Venn diagram, it is correct to conclude that  $\hat{\theta}_1(G)$  and  $\hat{\theta}_2(G)$  are key-equal.

$\boxed{2. \implies}$  From  $G \rightsquigarrow F$  and Lemma 2, it follows  $\text{vars}(G) \not\subseteq G^{+,q}$ . Since  $\text{key}(G) \subseteq G^{+,q}$ , we can assume a variable  $u \in \text{vars}(G) \setminus \text{key}(G)$  such that  $u \notin G^{+,q}$ . From the premise  $\hat{\theta}_1(G) = \hat{\theta}_2(G)$ , it follows  $\hat{\theta}_1(u) = \hat{\theta}_2(u)$ . Since  $x$  occurs in all regions outside  $G^{+,q}$  in the Venn diagram, we conclude  $\theta_1(x) = \theta_2(x)$ . Finally,  $\theta_1(y) = \theta_2(y)$  and  $\theta_1(z) = \theta_2(z)$  follow from item 1 proved before.  $\boxed{2. \impliedby}$  Trivial. This concludes the proof of Sublemma 3.  $\dashv$

**Proof Sublemma 4**  $\boxed{1.}$  Assume  $\mathbf{r}_0$  is a repair of  $\mathbf{db}_0$ . We first show that  $\text{map}(\mathbf{r}_0) \cap \mathbf{db}_F$  contains no two distinct key-equal facts. Let  $A, B \in \text{map}(\mathbf{r}_0) \cap \mathbf{db}_F$ . We can assume  $\theta_1, \theta_2 \in \mathcal{V}$  such that  $\theta_1(F_0), \theta_2(F_0) \in \mathbf{r}_0$ ,  $A = \hat{\theta}_1(F)$ , and  $B = \hat{\theta}_2(F)$ . By Sublemma 2, if  $A$  and  $B$  are key-equal and distinct, then  $\theta_1(F_0)$  and  $\theta_2(F_0)$  are key-equal and distinct, contradicting that  $\mathbf{r}_0$  is a repair. We conclude by contradiction that  $\text{map}(\mathbf{r}_0) \cap \mathbf{db}_F$  contains no distinct key-equal facts.

In an analogous way, one can use Sublemma 3 to show that

$\text{map}(\mathbf{r}_0) \cap \mathbf{db}_G$  contains no distinct key-equal facts. Since  $\mathbf{db}_{\text{rest}}$  is consistent, it follows that  $\text{map}(\mathbf{r}_0)$  is consistent.

We next show that  $\text{map}(\mathbf{r}_0)$  is a maximal consistent subset of  $\mathbf{db}$ . Let  $A \in \mathbf{db}_F$ . We need to show that  $\text{map}(\mathbf{r}_0)$  contains a fact that is key-equal to  $A$ . We can assume  $\theta \in \mathcal{V}$  such that  $A = \widehat{\theta}(F)$ . Since  $\mathbf{db}_0$  contains  $\theta(F_0)$  (by definition of  $\mathcal{V}$ ),  $\mathbf{r}_0$  contains  $\theta'(F_0)$  for some  $\theta' \in \mathcal{V}$  with  $\theta'(x) = \theta(x)$ . Consequently,  $\text{map}(\mathbf{r}_0)$  contains  $\widehat{\theta'}(F)$ . By Sublemma 2,  $A$  and  $\widehat{\theta'}(F)$  are key-equal. We conclude that  $\text{map}(\mathbf{r}_0)$  contains a fact that is key-equal to  $A$ .

In an analogous way, one can use Sublemma 3 to show that for every  $A \in \mathbf{db}_G$ ,  $\text{map}(\mathbf{r}_0)$  contains a fact that is key-equal to  $A$ .

**2.** Let  $\mathbf{r}$  be a repair of  $\mathbf{db}$ . Let  $\mathbf{r}_0$  be the following subset of  $\mathbf{db}_0$ .

$$\begin{aligned} \mathbf{r}_0 = & \quad \{\theta(F_0) \mid \widehat{\theta}(F) \in \mathbf{r}, \theta \in \mathcal{V}\} \\ & \cup \quad \{\theta(G_0) \mid \widehat{\theta}(G) \in \mathbf{r}, \theta \in \mathcal{V}\} \end{aligned}$$

We show  $\mathbf{r} = \text{map}(\mathbf{r}_0)$ . Since  $\mathbf{db}_{\text{rest}} \subseteq \mathbf{r} \cap \text{map}(\mathbf{r}_0)$ , it suffices to show  $\mathbf{r} \setminus \mathbf{db}_{\text{rest}} \subseteq \text{map}(\mathbf{r}_0)$  and  $\text{map}(\mathbf{r}_0) \setminus \mathbf{db}_{\text{rest}} \subseteq \mathbf{r}$ . Let  $A \in \mathbf{r} \setminus \mathbf{db}_{\text{rest}}$ . Let  $A = \widehat{\theta}(F)$ ,  $\theta \in \mathcal{V}$  (the case where  $A = \widehat{\theta}(G)$  is analogous). Then by definition of  $\mathbf{r}_0$ ,  $\theta(F_0) \in \mathbf{r}_0$ , hence  $\widehat{\theta}(F) \in \text{map}(\mathbf{r}_0)$ . Conversely, let  $A \in \text{map}(\mathbf{r}_0) \setminus \mathbf{db}_{\text{rest}}$ . Let  $A = \widehat{\theta}(F)$ ,  $\theta \in \mathcal{V}$  (the case where  $A = \widehat{\theta}(G)$  is analogous). By (3),  $\theta(F_0) \in \mathbf{r}_0$ . We can assume  $\theta' \in \mathcal{V}$  such that  $\widehat{\theta'}(F) \in \mathbf{r}$  and  $\theta'(F_0) = \theta(F_0)$ . By Sublemma 2,  $\theta'(F_0) = \theta(F_0)$  implies  $\widehat{\theta'}(F) = \widehat{\theta}(F)$ , hence  $A \in \mathbf{r}$ .

Using Sublemmas 2 and 3, it is straightforward to show that  $\mathbf{r}_0$  is a repair of  $\mathbf{db}_0$ .

**3.** Let  $\mathbf{r}_0, \mathbf{r}'_0$  be distinct repairs of  $\mathbf{db}_0$ . Then there exist distinct key-equal facts  $A, B$  such that  $A \in \mathbf{r}_0$  and  $B \in \mathbf{r}'_0$ . Assume  $A, B$  are  $R_0$ -facts (the case where  $A, B$  are  $S_0$ -facts is analogous). There exist valuations  $\theta_1, \theta_2 \in \mathcal{V}$  such that  $A = \theta_1(F_0)$  and  $B = \theta_2(F_0)$ . By Sublemma 2,  $\widehat{\theta_1}(F)$  and  $\widehat{\theta_2}(F)$  are distinct and key-equal. Since  $\widehat{\theta_1}(F) \in \text{map}(\mathbf{r}_0)$  and  $\widehat{\theta_2}(F) \in \text{map}(\mathbf{r}'_0)$ , and since  $\text{map}(\mathbf{r}_0), \text{map}(\mathbf{r}'_0)$  are consistent by property 1 shown earlier, it follows  $\text{map}(\mathbf{r}_0) \neq \text{map}(\mathbf{r}'_0)$ . This concludes the proof of Sublemma 4.  $\square$

## D Proofs of Section 6

**Proof Lemma 5** **1.** Let  $\tau$  be a join tree for  $q$ . Let  $\tau'$  be the graph obtained from  $\tau$  by replacing each vertex  $H$  with  $H_{[z \rightarrow c]}$ , and by adjusting edge labels (that is, every label  $L$  is replaced with  $L \setminus \{z\}$ ). Clearly,  $\tau'$  is a join tree for  $q'$ .

**2 and 3.** Let  $Q \subseteq q$ . Let  $X, Y \subseteq \text{vars}(Q)$ . Let  $Q' = Q_{[z \rightarrow c]}$ . In the next paragraph, we show that  $\mathcal{K}(Q) \models X \rightarrow Y$  implies  $\mathcal{K}(Q') \models X \setminus \{z\} \rightarrow Y \setminus \{z\}$ .

The computation of the attribute closure  $\{y \mid \mathcal{K}(Q) \models X \rightarrow y\}$  by means of a standard algorithm [1, page 165] corre-

sponds to constructing a maximal sequence

$$\begin{array}{rcl} X & = & S_0 \quad H_1 \\ & & S_1 \quad H_2 \\ & & \vdots \quad \vdots \\ & & S_{k-1} \quad H_k \\ & & S_k \end{array}$$

where

1.  $S_0 \subsetneq S_1 \subsetneq \dots \subsetneq S_{k-1} \subsetneq S_k$ ; and
2. for every  $i \in \{1, 2, \dots, k\}$ ,
  - (a)  $H_i \in Q$ . Thus,  $\mathcal{K}(Q)$  contains the functional dependency key( $H_i$ )  $\rightarrow$  vars( $H_i$ ).
  - (b) key( $H_i$ )  $\subseteq S_{i-1}$  and  $S_i = S_{i-1} \cup \text{vars}(H_i)$ .

Then,  $S_k = \{y \mid \mathcal{K}(Q) \models X \rightarrow y\}$ . Consider now the following sequence.

$$\begin{array}{rcl} X \setminus \{z\} & = & S_0 \setminus \{z\} \quad H_1_{[z \rightarrow c]} \\ & & S_1 \setminus \{z\} \quad H_2_{[z \rightarrow c]} \\ & & \vdots \quad \vdots \\ & & S_{k-1} \setminus \{z\} \quad H_k_{[z \rightarrow c]} \\ & & S_k \setminus \{z\} \end{array}$$

Clearly, for every  $i \in \{1, 2, \dots, k\}$ ,

1.  $H_{i[z \rightarrow c]} \in Q'$ . Thus,  $\mathcal{K}(Q')$  contains the functional dependency key( $H_{i[z \rightarrow c]}$ )  $\rightarrow$  vars( $H_{i[z \rightarrow c]}$ ).
2. key( $H_{i[z \rightarrow c]}$ )  $\subseteq S_{i-1} \setminus \{z\}$  and  $S_i \setminus \{z\} = (S_{i-1} \setminus \{z\}) \cup \text{vars}(H_{i[z \rightarrow c]})$ .

It follows that if  $\mathcal{K}(Q) \models X \rightarrow y$  and  $y \neq z$ , then  $\mathcal{K}(Q') \models X \setminus \{z\} \rightarrow y$ .

To prove 2, assume  $F \xrightarrow{q} G$ . Then, the unique path in  $\tau$  between  $F$  and  $G$  contains an edge with label  $L$  such that  $\mathcal{K}(q \setminus \{F\}) \models \text{key}(F) \rightarrow L$ . It follows  $\mathcal{K}(q' \setminus \{F'\}) \models \text{key}(F') \rightarrow L \setminus \{z\}$ . Since  $L \setminus \{z\}$  is a label on the unique path in  $\tau'$  between  $F'$  and  $G'$ , it follows  $F' \xrightarrow{q'} G'$ .

To prove 3, assume the attack  $F \xrightarrow{q} G$  is weak. Then  $\mathcal{K}(q) \models \text{key}(F) \rightarrow \text{key}(G)$ , hence  $\mathcal{K}(q') \models \text{key}(F') \rightarrow \text{key}(G')$ . It follows that the attack  $F' \xrightarrow{q'} G'$ , if it exists, is weak.  $\square$

**Proof Lemma 6** Assume each cycle in the attack graph of  $q$  is terminal. Assume towards a contradiction that three distinct atoms  $F, G, H$  of  $q$  belong to a same cycle such that  $F \rightsquigarrow G$  and  $G \rightsquigarrow H$ . By Lemma 3,  $F \rightsquigarrow H$  or  $G \rightsquigarrow F$ . In both cases, the attack graph contains a nonterminal cycle, a contradiction.  $\square$

**Proof Lemma 7** **1.** Let  $\tau$  be a join tree for  $q$ . By Lemma 6, every cycle in  $q$ 's attack graph is of the form

$H \rightsquigarrow H' \rightsquigarrow H$ . We show that if  $H \rightsquigarrow H' \rightsquigarrow H$ , then  $\tau$  contains an edge  $\{H, H'\}$ . Assume towards a contradiction that  $H \rightsquigarrow H' \rightsquigarrow H$  and there exists  $I \in q$  such that  $H \neq I \neq H'$  and  $I$  lies on the (unique) path in  $\tau$  between  $H$  and  $H'$ . From  $H \rightsquigarrow H'$ , it follows  $H \rightsquigarrow I$ . Then the cycle  $H \rightsquigarrow H' \rightsquigarrow H$  is not terminal, a contradiction.

Assume variable  $x$  occurs in two distinct cycles of  $q$ 's attack graph. We can assume disjoint cycles  $F \rightsquigarrow F' \rightsquigarrow F$  and  $G \rightsquigarrow G' \rightsquigarrow G$  such that  $x \in \text{vars}(F) \cup \text{vars}(F')$  and  $x \in \text{vars}(G)$ . Assume towards a contradiction  $x \notin \text{key}(F)$ . By the *Connectedness Condition*, if  $e$  is an edge on the unique path in  $\tau$  between  $F$  and  $G$  and  $e \neq \{F, F'\}$ , then the edge label of  $e$  contains  $x$ . Since the cycle  $F \rightsquigarrow F' \rightsquigarrow F$  is terminal,  $F \not\rightsquigarrow G$ . It follows  $x \in F^{+,q}$ . Since  $x \notin \text{key}(F)$ , we can assume an atom  $H \in q$  such that  $H \neq F$  and  $\text{key}(H) \subseteq \text{key}(F)$ . From  $F \rightsquigarrow F'$  and Lemma 2, it follows  $H \neq F'$ . By the premise of the lemma, we can assume  $H' \in q \setminus \{F, F'\}$  such that  $H \rightsquigarrow H' \rightsquigarrow H$ . By the *Connectedness Condition*, every edge label on the path in  $\tau$  between  $F$  and  $H$  contains  $\text{key}(H)$ . Let  $H \overset{L_1}{\rightsquigarrow} I_1 \overset{L_2}{\rightsquigarrow} I_2 \dots \overset{L_{m-1}}{\rightsquigarrow} I_{m-1} \overset{L_m}{\rightsquigarrow} I_m$  with  $I_m = F$  be the unique path in  $\tau$  between  $H$  and  $F$ . Two cases can occur.

**Case  $I_1 = H'$ .** From  $H' \not\rightsquigarrow I_2$ , it follows  $L_2 \subseteq H'^{+,q}$ . Since  $\text{key}(H) \subseteq L_2$ ,  $\text{key}(H) \subseteq H'^{+,q}$ .

**Case  $I_1 \neq H'$ .** Since  $H' \rightsquigarrow H$  and  $H' \not\rightsquigarrow I_1$ , it follows  $L_1 \subseteq H'^{+,q}$ . Since  $\text{key}(H) \subseteq L_1$ ,  $\text{key}(H) \subseteq H'^{+,q}$ .

Hence,  $\text{key}(H) \subseteq H'^{+,q}$ . Then by Lemma 2,  $H' \not\rightsquigarrow H$ , a contradiction. We conclude by contradiction  $x \in \text{key}(F)$ .

[2.] Assume  $F \overset{q}{\rightsquigarrow} G$  is a weak attack. Let  $x \in \text{key}(G) \setminus \text{key}(F)$ . By property 1 proved above, for all  $H \in q \setminus \{F, G\}$ ,  $x \notin \text{vars}(H)$ . Since the attack  $F \overset{q}{\rightsquigarrow} G$  is weak,  $\mathcal{K}(q) \models \text{key}(F) \rightarrow x$ . Since  $x$  does not occur in  $q \setminus \{F, G\}$ , it must be the case that  $x \in \text{vars}(F)$ . It follows  $\text{key}(G) \subseteq \text{vars}(F)$ .  $\square$

**Proof Lemma 8** Assume that  $F$  is an  $R$ -atom. Since  $\mathbf{db}$  is purified relative to  $q$ , we can assume that the set of  $R$ -facts of  $\mathbf{db}$  is  $\mathcal{R} = \{F_{[\vec{y} \rightarrow \vec{b}_1]}, \dots, F_{[\vec{y} \rightarrow \vec{b}_\ell]}\}$  for some  $\ell \geq 0$  and  $\vec{b}_1, \dots, \vec{b}_\ell \in D^{|\vec{y}|}$ . Since  $\text{key}(F) = \emptyset$ , all facts of  $\mathcal{R}$  are key-equal.

[1  $\implies$  2] Since  $\mathbf{db} \models q$ , we have  $\ell \geq 1$ , hence  $\mathbf{db} \neq \emptyset$ . Let  $\mathbf{r}$  be a repair of  $\mathbf{db}$  and  $i \in \{1, \dots, \ell\}$ . We need to show that  $\mathbf{r} \models q'_{[\vec{y} \rightarrow \vec{b}_i]}$ . Since  $(\mathbf{r} \setminus \mathcal{R}) \cup \{F_{[\vec{y} \rightarrow \vec{b}_i]}\}$  is a repair of  $\mathbf{db}$ , it follows from the premise that  $(\mathbf{r} \setminus \mathcal{R}) \cup \{F_{[\vec{y} \rightarrow \vec{b}_i]}\} \models q$ . Since  $q$  contains no self-join, it follows  $(\mathbf{r} \setminus \mathcal{R}) \models q'_{[\vec{y} \rightarrow \vec{b}_i]}$ , hence  $\mathbf{r} \models q'_{[\vec{y} \rightarrow \vec{b}_i]}$ .

[2  $\implies$  1] Let  $\mathbf{r}$  be a repair of  $\mathbf{db}$ . By the premise, for every  $j \in \{1, \dots, \ell\}$ ,  $\mathbf{r} \models q'_{[\vec{y} \rightarrow \vec{b}_j]}$ . We can assume  $i \in \{1, \dots, \ell\}$  such that  $F_{[\vec{y} \rightarrow \vec{b}_i]} \in \mathbf{r}$ . From  $\mathbf{r} \models q'_{[\vec{y} \rightarrow \vec{b}_i]}$ , it follows  $\mathbf{r} \models q$ .  $\square$

**Proof Sublemma 5** [1  $\implies$  2] Let  $\mathbf{r} = \mathbf{r}_1 \cup \mathbf{r}_2$  be a repair of  $\mathbf{db}$  such that for all  $i \in \{1, \dots, \ell\}$ , for all partitions  $P$  of  $\mathbf{db}_i$ ,

- if  $P \notin \text{CERTAINTY}(q_i)$ , then  $\mathbf{r}_1$  contains a repair of  $P$  falsifying  $q_i$ ; and
- if  $P \in \text{CERTAINTY}(q_i)$ , then  $\mathbf{r}_2$  contains a repair of  $P$ .

Since  $\mathbf{db} \in \text{CERTAINTY}(q)$  by the premise, we have  $\mathbf{r} \models q$ . For all  $i \in \{1, \dots, \ell\}$ , for every valuation  $\theta$ , if  $\theta(q_i) \subseteq \mathbf{r}$ , then  $\theta(F_i), \theta(G_i)$  must belong to the same partition of  $\mathbf{db}_i$ , hence  $\theta(F_i), \theta(G_i) \subseteq \mathbf{r}_2$ . Consequently,  $\mathbf{r}_2 \models q$ . Since  $\mathbf{r}_2 \subseteq \bigcup_{1 \leq i \leq \ell} \llbracket \mathbf{db}_i \rrbracket$ , we have  $\bigcup_{1 \leq i \leq \ell} \llbracket \mathbf{db}_i \rrbracket \models q$ .

[2  $\implies$  1] Let  $\mathbf{r}$  be a repair of  $\mathbf{db}$ . Let  $\vec{x}$  be a sequence of distinct variables containing every variable  $x$  such that for some  $1 \leq i < j \leq \ell$ ,  $x \in \text{vars}(q_i) \cap \text{vars}(q_j)$ . By the premise, we can assume  $\vec{a} \in D^{|\vec{x}|}$  such that  $\bigcup_{1 \leq i \leq \ell} \llbracket \mathbf{db}_i \rrbracket \models q_{[\vec{x} \rightarrow \vec{a}]}$ . Let  $\theta$  be the valuation over  $\text{vars}(\vec{x})$  such that  $\theta(\vec{x}) = \vec{a}$ . For  $1 \leq i \leq \ell$ ,  $\text{vars}(\vec{x}_i) \subseteq \text{vars}(\vec{x})$ . For  $1 \leq i \leq \ell$ , define  $\vec{a}_i \in D^{|\vec{x}_i|}$  as the sequence of constants such that  $\vec{a}_i = \theta(\vec{x}_i)$ . Then, for each  $i \in \{1, \dots, \ell\}$ ,  $\llbracket \mathbf{db}_i \rrbracket \models q_{i[\vec{x}_i \rightarrow \vec{a}_i]}$ . Since  $\llbracket \mathbf{db}_i \rrbracket$  contains a partition with vector  $\vec{a}_i$  which belongs to  $\text{CERTAINTY}(q_i)$  (by construction), every repair of  $\llbracket \mathbf{db}_i \rrbracket$  satisfies  $q_{i[\vec{x}_i \rightarrow \vec{a}_i]}$ . Since  $\mathbf{r}$  contains a repair of  $\llbracket \mathbf{db}_i \rrbracket$ , we conclude  $\mathbf{r} \models q_{i[\vec{x}_i \rightarrow \vec{a}_i]}$ . Since  $\vec{x}$  contains all variables that occur in two distinct queries among  $q_1, \dots, q_\ell$ , it is correct to conclude  $\mathbf{r} \models q_{[\vec{x} \rightarrow \vec{a}]}$ . Since  $\mathbf{r}$  is an arbitrary repair of  $\mathbf{db}$ ,  $\mathbf{db} \in \text{CERTAINTY}(q)$ .  $\dashv$

**Proof Lemma 9** Let  $\mathcal{D}$  be the set of uncertain databases. We define a mapping  $f : \mathcal{D} \rightarrow \mathcal{D}$  as follows. If  $\mathbf{db}$  is an uncertain database with active domain  $D$ , then  $f(\mathbf{db})$  is the smallest set such that

- for every fact  $A \in \mathbf{db}$ , if the relation name of  $A$  occurs in  $q'$ , then  $A \in f(\mathbf{db})$ ; and
- whenever  $R(\vec{x})$  in  $q \setminus q'$  and  $\vec{a} \in D^{|\vec{x}|}$ , then  $f(\mathbf{db})$  contains  $R(\vec{a})$ .

It is straightforward that  $f$  is first-order expressible and  $\mathbf{db} \in \text{CERTAINTY}(q') \iff f(\mathbf{db}) \in \text{CERTAINTY}(q)$ .  $\square$

**Proof Corollary 1** By Lemma 9, there exists an  $\text{AC}^0$  many-one reduction from  $\text{CERTAINTY}(\text{C}(k))$  to  $\text{CERTAINTY}(\text{AC}(k))$ . Then by Theorem 4,  $\text{CERTAINTY}(\text{C}(k))$  is in  $\mathbf{P}$ .  $\square$

## E Proofs of Section 7

**Proof Proposition 1** [1  $\implies$  2] Let  $\mathbf{w} \in \text{worlds}(\mathbf{db})$  such that  $\text{Pr}(\mathbf{w}) > 0$ . We need to show  $\mathbf{w} \models q$ . From  $\text{Pr}(\mathbf{w}) > 0$ , it follows that for every block  $\mathbf{b}$  of  $\mathbf{db}$ , if  $\sum_{A \in \mathbf{b}} \text{Pr}(A) = 1$ , then  $\mathbf{w}$  contains a fact of  $\mathbf{b}$ . Consequently, there exists a repair  $\mathbf{r}$  of  $\mathbf{db}'$  such that  $\mathbf{r} \subseteq \mathbf{w}$ . Since  $\mathbf{r} \models q$  by the premise, it follows  $\mathbf{w} \models q$ .

[2  $\implies$  1] Let  $\mathbf{r}$  be a repair of  $\mathbf{db}'$ , hence  $\mathbf{r} \in \text{worlds}(\mathbf{db})$ . We have  $\text{Pr}(\mathbf{r}) > 0$ , because for every block  $\mathbf{b}$  of  $\mathbf{db}$ , if  $\sum_{A \in \mathbf{b}} \text{Pr}(A) = 1$ , then  $\mathbf{r}$  contains a fact of  $\mathbf{b}$ . By the premise,  $\mathbf{r} \models q$ .  $\square$

**Proof Corollary 2** Assume  $\text{CERTAINTY}(q)$  is not first-order expressible. By Theorem 6, query  $q$  is not safe. By Theorem 5,  $\text{PROBABILITY}(q)$  is  $\sharp\mathbf{P}$ -hard.  $\square$